Studies in Astronomical Time Series Analysis. VI. Optimal Segmentation: Blocks, Histograms and Triggers

Jeffrey D. Scargle

Space Science Division, NASA Ames Research Center

Jay Norris

Laboratory for High Energy Astrophysics
Code 661, NASA Goddard Space Flight Center

Brad Jackson

San José State University,
Department of Mathematics and Computer Science,
The Center for Applied Mathematics and Computer Science

Draft of January 28, 2005.

Abstract

Detecting and characterizing statistically significant signals in noisy time series data can be achieved by optimizing a piecewise constant model. A surprisingly simple algorithm implements this idea by finding the partition of the observation interval that maximizes a model fitness function. We present the algorithm and fitness functions for a variety of data modes, including points, binned points, and measurements at arbitrary times and with normal errors. This exact algorithm has an inductive structure that allows either real-time or retrospective analysis of time series. Examples are given to demonstrate time series analysis, real-time triggers, and optimal histograms with bin size and location determined by the data, not arbitrarily fixed ahead of time.

Key words: time series, data analysis, non parametric models, density estimation, histograms

Contents

1 Introduction: Block Segmentation

We present an improved and generalized version of *Bayesian Blocks* [Scargle 1998], a method of detecting and characterizing variability – both random and deterministic [Scargle 1981] – in time series data corrupted by observational errors. To start with, we introduce a simple abstract representation appropriate for any sequential data. Further, an exact method to find the global optimum partition of an interval replaces the "greedy" algorithms of earlier work. And finally, we develop new cost functions for a variety of data modes and based on maximized likelihoods and posterior probability distributions. The resulting algorithms are meant for exploratory data analysis and automated data understanding. As will be discussed in a subsequent publication, this entire framework is easily generalized to data of higher dimensionality, such as images, photon maps, redshift surveys and the like.

A key goal is to impose as few prior conditions on the signal as possible. In particular, we wish to avoid smoothness or shape assumptions that place *a priori* limitations on scales and resolution. The algorithm should handle arbitrary sampling (*i.e.*, not limited to evenly spaced, gapless data) and large dynamic ranges in amplitude and scale. For scientific data mining applications and for objectivity, the method should be automatic. It should eliminate noise as much as possible, while conserving most of the valid information in the data. It should be applicable to multivariate problems. Incorporation of auxiliary, extrinsic data, such as spectral or color information, and variable exposure, should be possible. It should be able to operate both retrospectively (model of all the data after it is collected) and in a real-time fashion that triggers on the first significant variation of the signal from its initial value.

Our algorithm achieves these desiderata in a simple computational framework that is easy to use and represents the signal structure in a form handy for further analysis, including the estimation of derivative quantities of astrophysical interest. It includes an automatic penalty for model complexity, thus solving the vexing problem often called *determining the order of the model.* It is exact, not a *greedy approximation*[1] as in [Scargle 1998].

Its limitations include that it detects local, rather than global, structure, and while much faster than an explicit search of the exponentially large parameter space, the runtime of the simple algorithm is $O(N^2)$. This computational complexity is considered prohibitive in some large problems, but an effective way to reduce the time to $\sim NlogN$ is in development.

These desiderata suggest the use of the most generic possible nonparametric data model, and have motivated our development of data segmentation and Bayesian changepoint methods [Ò Ruanaidh and Fitzgerald 1996]. It is remarkable that a very simple idea – *fitting of piecewise constant models to the data* – achieves essentially all of the above desiderata. This approach yields a step-function, or segmented, representation of the signal in which the range of the independent variable (*e.g.* time) is automatically divided into unequal subintervals, in each of which the dependent variable (*e.g.* intensity) is modeled as constant.

2 The Model: Piecewise Constant

As just indicated we are led to employ a very simple model, in which the data interval is partitioned into segments (here called *blocks*) and

[1] An iteration making an optimal improvement at each step, but not guaranteed of an optimal overall solution.

the signal is taken to be constant within each segment. The model has three parameters for each block: the start time, the duration, and the signal amplitude (*e.g.*, the Poisson rate parameter for event data). In the model of the full data interval the first two parameters are not independent: one block begins where another leaves off. Specifically, we represent these parameters in terms of a finite set of changepoints, essentially one per block.

This representation is in the spirit of a nonparametric approximation, and not meant to imply that we believe the signal is actually discontinuous. The crude, blocky appearance of our discontinuous model may be a liability in the context of visualization, but for our interests in deriving physically meaningful quantities we have not found it so. Blocky models are useful in broad signal processing contexts [Donoho 1994], and have several motivations. Their simplicity allows exact treatment of the likelihood. We can optimize or marginalize the rate parameters exactly, giving simple formulas for the fitness function. And we regard the estimated model itself as less important than quantities derived from it. For example, while smoothed plots of pulses within gamma-ray bursts make pretty pictures, one is really interested in pulse locations, lags, amplitudes, widths, rise and decay times, *etc.* These quantities can be accurately determined directly from the locations, heights and widths of the blocks.

Especially for applications in measuring similarity among time series and pattern matching, piecewise linear models are often used (*cf* the work of Heikki Mannila and Eamonn Keogh). Such models may have a better visual appearance, but in our experience the improved flexibility is largely offset by added complexity of the model and its interpretation. Note further that if continuity is imposed at the changepoints, a piecewise linear model has essentially the same number of parameters, or degrees of freedom, as does the simpler piecewise constant model.

Below §3 discusses partitions of the data interval, a convenient data representation scheme, and the new algorithm for computing optimal partitions. Then §4 exhibits the computation of cost functions for a variety of data modes, followed by numerical simulations and other examples in §5.

3 Optimum Partition of an Interval

Our algorithm works on any sequential data. We introduce it in a somewhat abstract setting because it can be used for other partitioning problems beyond time series analysis. In a special case it implements Bayesian blocks or other 1D segmentation ideas for any model fitness function that satisfies a simple additivity condition. It improves on our previous approximate segmentation algorithms by achieving a rigorous solution of the multiple changepoint problem, and is guaranteed to find the global maximum, not just a local one. This is made possible by reducing the infinite search space to the finite set of partitions consisting of blocks composed of discrete data cells.

3.1 Data Cells

The set of possible values of the independent variable is called the *data space*. For the one dimensional case treated here, the data space is usually an interval, such as the time over which observations have been made. The measured quantity can be almost anything. Most commonly it is either a physical variable or the density of discrete events.

Consider observational data comprising N sequential elements

$$\mathbf{x}_n, \quad n = 1, 2, \ldots, N. \tag{1}$$

The specific meaning of the quantities $\mathbf{x}_n$ is left vague because almost any of a wide variety of data types can be treated within this formalism. Simple examples are: points, counts of points in bins, and measurements – correspondingly, the array $\mathbf{x}$ would contain point coordinates; counts, bin sizes and locations; and measured values and their uncertainties, respectively. The only requirement is that the data be ordered (*i.e., sequential*), meaning that each $\mathbf{x}_n$ is associated with a time t_n, such that the latter are ordered and contained in some time interval I:

$$min(I) \leq t_1 < t_2 < \ldots < t_N \leq max(I) \ \ . \tag{2}$$

In general t_n specifies the time of measurement, be it a point or an interval. For event data (also called point data), t_n is just the time of event n. Although times are often represented as real numbers, the finite accuracy of measurement means that one is really specifying an integer multiple of some small unit of time (typically on the order of milliseconds to microseconds in high energy astrophysics). For cases such as binned counts or measurements averaged over finite time intervals, the time interval must be specified, either explicitly (as in an array giving the lengths of a series of unequal time bins) or implicitly (*e.g.* through specification of bin size and time of the first bin).

It is convenient to represent sequential data with a data structure consisting of a set of N *data cells*

$$C_n \equiv \{\mathbf{x}_n, t_n\} \ n = 1, 2, \ldots N \ , \tag{3}$$

They form an ordered sequence with respect to the independent variable t, can be grouped into blocks (§3.2) forming partitions of I (§3.3), and contain whatever data quantities are necessary to evaluate the fitness (§3.4) of an arbitrary partition. In some cases two or more data elements are combined into a single cell (see *e.g.* the discussion of duplicate time tags in §4.1), but for the most part data cells correspond one-to-one with data elements. In some cases (*e.g.* time-tagged event data) t_n is contained in $\mathbf{x}_n$ and need not be separately specified. Figure 1 is a cartoon of typical data cells.

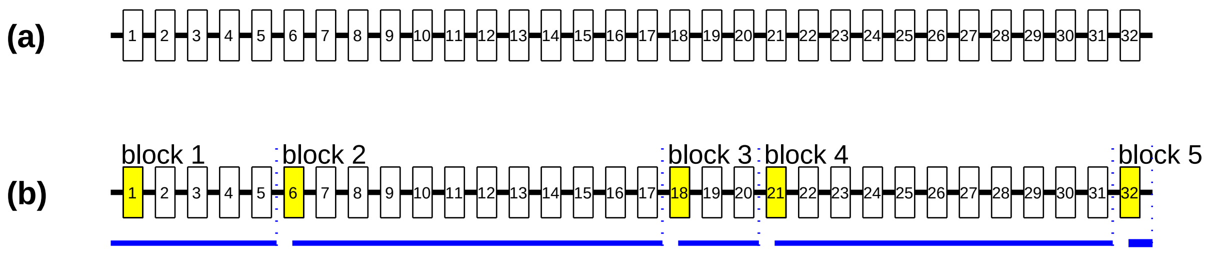

Figure 1: Pictorial representation of data cells and the blocks made from them. The horizontal axis represents the independent variable (often, but not necessarily time), with respect to which the data are ordered. The sequential order depicted in Panel (a) is the only essential requirement for data to be analyzable with our block algorithm. Panel (b) exemplifies the partition of the set of data cells into blocks. The shaded cells are changepoints marking the beginnings of the blocks.

3.2 Blocks of Cells

A block is a set of adjacent cells. Panel (b) of Figure 1 shows a sequence of 32 data cells divided into five blocks. The following notation for blocks is useful:

$$\mathbf{B}(n,m) \equiv \{C_n, C_{n+1}, \ldots C_m\} \ , \tag{4}$$

that is $m - n + 1$ cells in sequence. The case $m = n$ represents a block consisting of just one cell, as in the last block of the partition in Figure 1(b). The model of the time series data is segmented into blocks, meaning that any model parameters are constant within each block but undergo discrete jumps at the changepoints (§3.5) marking the edges of the blocks. The fitness of a block is of elementary importance, because the fitness of a partition (§3.4) is the sum of the fitness of the blocks comprising it.

3.3 Partitions

A *partition* of the interval I is simply a set of non-overlapping blocks that together add up to the whole interval.[2] A partition can be defined by specifying the number of blocks (the elements of the partition) N_{blocks}, and the block edges n_k:

$$\mathcal{P}(I) \equiv \{N_{blocks},\ n_k,\ \ k = 1, 2, 3, \ldots N_{blocks}\}\ . \tag{5}$$

There are one fewer changepoints than blocks, since by convention the first block begins at the first data cell – $n_1 \equiv 1$ is implicit – and the last block terminates with the last data cell. As described in §3.4 we will seek the partition that maximizes a given function over all possible partitions. How big is this search space if there are N cells? Establish a 1-1 mapping between partitions and binary numbers of length N, by setting the k-th digit to 1 if cell k is a changepoint, 0 otherwise. Remembering that the first cell is always a changepoint, the number of partitions is then

$$N_{\text{partitions}} = 2^{N-1} \tag{6}$$

[2]Formally a partition of I is a set of blocks satisfying $I = \bigcup_k B_k$ and $B_j \bigcap B_k = \emptyset$ (the null set), for $j \neq k$.

Except for short time series this number is too large for an exhaustive search, but our algorithm nevertheless finds the optimum over this space in a time that scales as only N^2.

3.4 Fitness of a Partition

Since our goal is to model data, we maximize[3] a quantity measuring the fitness of models in a specified class. We take as this model class all partitions of the interval, with a given statistical model for each block of the partition. If the observational errors at different times are independent, as is often the case, fitness is additive over blocks:

$$F[\mathcal{P}(I)] = \sum_{k=1}^{N_{blocks}} f(B_k) \quad , \tag{7}$$

where $F[\mathcal{P}(I)]$ is the total fitness and $f(B_k)$ is the fitness of block k. Our algorithm depends explicitly on this additivity.

Specific examples and details of fitness functions are given below in §4. What is important here is that we marginalize, or otherwise eliminate, all parameters of the block models except the times defining the beginning and end of the block (Paper V). Then the total fitness depends on only $\mathcal{P}(I)$. The best model is found by maximizing F over all partitions. As an example, the fitness function we adopt for count data does not depend on the Poisson rate parameters – they can be computed in an almost trivial way, once the changepoints of the optimum partition are determined.

[3]Alternatively, one can minimize an error measure. Both are called *optimization*.

3.5 Changepoints

We call the time separating two blocks a *changepoint*[4]. In principle a changepoint could be anywhere in the interval, but we restrict them to occur at the times corresponding to the data cells. The reasoning is that moving a changepoint lying between two data cells to a new location between the same cells does not sensibly change the model's representation of the data. This simplification reduces the search over an infinite space to a finite optimization problem.

In some applications it might be useful to assign a data cell that is a changepoint to be in *both* the subsequent and previous blocks, but here we assign it to only one – with the convention that a changepoint is the first cell in the subsequent block (rather than the last cell of the previous block). Correspondingly, since the smallest partition consists of a single block containing all data cells, the first data cell is always a changepoint. If the last cell is a changepoint, it demarcates a block consisting of that one cell, as in panel (b) of Figure 1, where the five changepoints dividing the data cells into five blocks are shaded.

3.6 A Lemma on Subpartitions

We define a *subpartition* of a given partition $\mathcal{P}(I)$ is a partition (of a subset of I) consisting of a subset of the blocks of $\mathcal{P}(I)$. Although not a necessary condition for the lemma to be true, in all cases of interest here the blocks in the subpartition are contiguous, and thus form a partition of a subinterval of I. Below we will make use of this simple result on subpartitions of optimal partitions:

[4]In statistics, a changepoint in a time series is a point at which the statistical model undergoes an abrupt transition, usually by one or more of its parameters jumping to a new value

Lemma: A subpartition of an optimal partition is an optimal partition of the subset it covers.

Let $\mathcal{P}'$ be the subpartition and I' the subset of I that it covers. If there were a partition of I', different from and fitter than $\mathcal{P}'$, then combining it with the blocks of $\mathcal{P}$ not in $\mathcal{P}'$ would, by the block additivity condition, yield a partition of I fitter than $\mathcal{P}$, contrary to the optimality of $\mathcal{P}$. ■

Corollary: removing the last block of an optimal partition leaves an optimal partition.

3.7 The Algorithm

We have assembled the definitions and results needed to state our procedure and prove that it finds a global optimum partition. This algorithm is in the spirit of dynamic programming [Hubert, Arabie, and Meulman 2001]. It begins with the first data cell, adding one more at each step until the whole interval has been treated. This feature makes the algorithm suitable for real-time applications (see §5.7).

The proof is by mathematical induction: if a theorem is true for $R = 1$, and one can show that, if it is true for R then it is true for $R+1$, then the theorem holds for all R. At step R the algorithm finds the optimum partition of the interval comprised of data cells $I_R \equiv \{C_1, C_2, \ldots C_R\}$. To analyze all the data, take $R = 1, 2, \ldots N$. The case $R = 1$ is trivial: there is only one cell, and the only partition possible is the optimum one.

Now suppose we have completed step R, having obtained the optimal partition $\mathcal{P}^{opt}[I_R]$, hereafter abbreviated $\mathcal{P}^{opt}(R)$, and are now at step $R+1$ and wish to find the optimal partition $\mathcal{P}^{opt}$(R+1). Assume further that we have kept a running record of the fitness of the optimum partition obtained at each previous step (call this array `best`) and the location of the last changepoint in that partition (call this array `last`). It is straightforward to compute

$$M(r) \equiv f[\mathbf{B}(r, R+1)] \quad (r = 1, 2, \ldots R+1) \tag{8}$$

that is, the fitness of a putative last block starting at r and extending to the end of the current interval. For example $M(1)$ is the fitness of the whole interval currently in play, namely the cells from 1 through $R+1$.

Using the block additivity of fitness, Eq. (7), the fitness of the partition of I_{R+1} consisting of the optimum partition $\mathcal{P}^{opt}[I_{r-1}]$ followed by a single block $\mathbf{B}(r, R+1)$ is:

$$A(r) = M(r) + \left\{ \begin{array}{ll} 0 & r = 1 \\ \texttt{best}(r-1), & r = 2, 3, \ldots, R+1 \end{array} \right. , \tag{9}$$

Now comes the key reasoning step. While we don't yet know what it is, the new optimum partition $\mathcal{P}^{opt}(R+1)$ must exist and must have a *last changepoint*, say r^*.[5] From its definition $A(r^*)$ is the fitness of $\mathcal{P}^{opt}(R+1)$. In particular, $\texttt{best}(r^*-1)$ is the fitness of the optimal subpartition consisting of all but the last block of $\mathcal{P}^{opt}(R+1)$, and $M(r*)$ is the fitness of said last block. Further, any partition with its last changepoint at some other $r \neq r^*$ must have fitness not greater than that of $\mathcal{P}^{opt}(R+1)$, so we have

$$A(r) \leq A(r^*) \quad \text{for } r \neq r^* \ . \tag{10}$$

In other words, the maximum of $A(r)$ occurs at r^*:

$$r^* = \text{argmax}[A(r)] \ , \tag{11}$$

[5]Any finite combinatorial optimization problem has at least one solution. Also, all partitions have at least one changepoint.

so finding the fitness and last changepoint of $\mathcal{P}^{opt}(R+1)$ is just a matter of finding the maximum of the array A and the index r at which this maximum occurs.

At the end of the computation, it only remains to find the locations of the optimal changepoints. The needed information is contained in the array $\texttt{last}(r)$ in which we have stored the index $r*$ at each step. Using the corollary of the subpartition lemma, it is a simple matter to use the last value in this array to determine the last changepoint in $P^{opt}(N)$, peel off the end section of `last` corresponding to this last block, and repeat. That is to say, the values

(1) $cp_1 = \texttt{last}(N)$

(2) $cp_2 = \texttt{last}(cp_1 - 1)$

(3) $cp_3 = \texttt{last}(cp_2 - 1)$

$\ldots$

are the index values giving the locations of the changepoints, in reverse order. The positions of the changepoints are not necessarily fixed until the very last iteration, although in practice it turns out that they become more or less "frozen" once a few succeeding changepoints have been detected.

The MatLab code for the algorithm in Appendix XX indicates how all of these computations are implemented.

4 Block Fitness Functions for Sequential Data

Here we outline the computation of *model fitness*. The *fitness function* for a fixed block of data numerically evaluates how well a constant signal strength represents whatever data lie in that block. The resulting quantities for the blocks covering the data interval are combined to form a fitness measure for the complete piecewise constant model.

For our algorithm to work, the total model fitness must depend on only parameters which specify the locations of the block edges, *i.e.* the changepoints. We must first account for, and eliminate, all other parameters. In the basic model the parameters specifying the signal strength for each block must be eliminated. The simplest way to do this is to take block fitness to be the likelihood maximized over all values of the signal strength. Another approach is to treat signal strength as a nuisance parameter and marginalize it. In the latter case fitness is not regarded as an absolute goodness-of-fit, but rather as a Bayes factor. Both approaches yield a quantity assessing alternative piecewise constant models for the fixed data.

Computation of fitness functions may be very different from case to case, but the following features are common to all of the data modes considered in this paper.

In all cases the fitness function depends only on the parameters defining the error distribution of whatever measurements lie in the block. For event data governed by the Poisson distribution (§§4.1, 4.1.4), there are exactly two such sufficient statistics: N, the number of events in the block, and M, the length of the block. In other cases (*e.g.* §4.2) the number of parameters depends on the form of the distribution. In all cases treated here, the sufficient statistics for a block are the sums of those for its cells. This relation simplifies the computation, but is not essential to the algorithm.

Furthermore, the sufficient statistics are local quantities, defined only for individual cells or blocks. In a sense we can ignore the actual locations of the cells assigned to a block; In principle, the cells in a block need not even be contiguous. This for example allows a very simple treatment of data gaps. Or the cells near the beginning and the end of the interval might be assigned to the same block–for example, the pre-burst and post-burst data from a gamma ray burst could combine into a single block representing a constant background. An algorithm explicitly allowing wraparound would be a natural way to deal with this case. However, for most purposes it is convenient to impose cell contiguity on the blocks, and our algorithm has this condition built in.

Finally, there are two types of factors in a fitness function that can be ignored, for different reasons. First, a factor in the likelihood for each data cell that does not depend on the rate parameter yields a simple constant factor for the whole time series (namely the product of the factor over all the data cells), independent of both the rate parameter and where the changepoints lie. Such a factor cancels out in any explicit model comparison, and is irrelevant as well for the implicit model comparison that takes place in our optimization algorithm. Second, there are factors in the fitness function for each block that are independent of the rate parameter. These factors do matter, but they contribute to the log of the fitness function a term proportional to the number of blocks, and as such can be absorbed into the parameter derived from the prior on the number of blocks (*cf.* §4.5).

4.1 Event Data

Sometimes the physical process, or perhaps the way it is recorded, takes the form a sequence of discrete events, each yielding a point in the data space. (In practice, the coordinates of the points are integer multiples of some small but finite unit–and are thus discrete, not continuous. This fact is important for the computations below.) The quantity of ultimate interest is the *distribution function* of the points, interpretable as the intensity or probability density of some physical variable. Accordingly the terms *density estimation* and *rate estimation* are sometimes used. A key example is the case where the events are the detection of individual photons, the corresponding points are the measured detection times, and the quantity of interest is the radiation intensity as a function of time.

For point data, it is natural to associate one cell with each event. However, if the detector can detect two (or more) events that are simultaneous to within its timing accuracy, such pairs would be assigned to the same cell. Since data cells must contain whatever information is necessary to compute the fitness function of a block containing the cells (§3.1), the data structure representing the cells must contain the number of events assigned to the cell (most often 1) and the length of the interval associated with the event.

There is more than one way to make such an association between sequential events and intervals. Perhaps most natural is to assign to a point *all times closer to it than to any other data point.* This resulting intervals join the midpoints between successive events. This concept generalizes to data spaces of any dimensions (where it is called the the *Voronoi tessellation* of the data points, [Okabe, Boots, Sugihara and Chiu 2000, Scargle 2001a, Scargle 2001c]), allowing finite partitions which adequately approximate the infinite set of arbitrary partitions.

Alternatively, one can use the intervals between successive data points–assigning half of an event to the interval immediately to its left and half to the one immediately to its right. This choice may handle the onset of a steep gradient in the underlying density slightly better, and is also easily generalized to higher dimension where it is known as the *Delaunay triangulation* [Okabe, Boots, Sugihara and Chiu 2000]. The algorithm described below allows use of either of these interval schemes.

The analysis in §2.2.1 of [Scargle 1998] can be carried over largely unchanged to the cell-based approach described here. But we offer several extensions of that work. First, we develop a new class of fitness functions based on maximizing the likelihood with respect to the rate parameter, in contrast to marginalizing it as in computing the Bayesian posterior. In addition, for the case where we compute the posterior we consider a prior with a finite range, as opposed to the flat prior over an infinite range. And we include variable bin size and exposure factors.

As mentioned above, and detailed in §2.2.1 of [Scargle 1998], assume that there is an elementary quantum of time–a *tick*–set by the measurement system. This is the finest time resolution the measurement apparatus is capable of recording. Let n_m be the number of events (*e.g.* photons) detected in tick m. We consider two data modes. In mode 1 the number of events in a given tick is presumed to follow a Poisson distribution. Mode 2 corresponds to situations where detection of more than one event at a given time is not possible, typically due to the *deadtime* of the detector, so that the number of events in a tick can be only 0 or 1. An example is time-tagged event (TTE) data in which duplicate time tags are not allowed. The fitness functions for the two modes, while similar, are different enough that the appropriate one should be used in practice.

4.1.1 Poisson Distributed Event Data

For mode 1, the likelihood for tick m is, from the Poisson distribution

$$L_m = \frac{\lambda^{n_m} \ e^{-\lambda}}{n_m!} \ . \tag{12}$$

The block likelihood is the usual product

$$L^{(k)} = \prod_{m=1}^{M^k} \frac{\lambda^{n_m} e^{-\lambda}}{n_m!} \ . \tag{13}$$

where M^k is the number of ticks in block k. Simplifying and collecting the factors for ticks with the same number of events, we have

$$L^{(k)} = e^{-\lambda M^k} \prod_{n=0}^{\infty} (\frac{\lambda^n}{n!})^{H(n)} \ , \tag{14}$$

where $H(n)$ is the number of ticks in the block with n events. The factor resulting from the factorial in the denominator is a constant, independent of the model, and therefore irrelevant for model comparison. Dropping this factor, and noting that $\Sigma_{n=0}^{\infty} nH(n) = N^k$, we have

$$L^{(k)} = \lambda^{N^k} e^{-\lambda M^k} \tag{15}$$

In this context it is often suggested that one should employ the intervals between successive events, since they in some sense carry the rate information more directly than do the actual times. We will now show that the likelihood based on intervals is essentially equivalent to the one above. It is a classic result [Papoulis 1965] that intervals between independent events distributed uniformly in time with a constant rate λ is exponential:

$$P(dt) = \lambda e^{-\lambda dt} U(dt), \tag{16}$$

where $U(x)$ is the unit step function:

$$\begin{aligned} U(x) &= 1 \qquad x \geq 0 \\ &= 0 \qquad x < 0 \end{aligned}$$

Pretend that the data consists of the inter-event intervals, and we do not even know the absolute times. The likelihood of our constant-rate Poisson model for interval $dt_n \geq 0$ is

$$L_n = \lambda e^{-\lambda dt_n}, \tag{17}$$

so the block likelihood is

$$L^{(k)} = \prod_{n=1}^{N^k} \lambda e^{-\lambda dt_n} = \lambda^{N^k} e^{-\lambda M^k}, \tag{18}$$

This likelihood is the same as that in Eq. (15).

There are two ways to proceed. The first is to find the maximum of this likelihood as a function of λ, which is at $\lambda = \frac{N^k}{M^k}$, so we have

$$L_{max} = (\frac{N^k}{M^k})^{N^k} e^{-N^k} \tag{19}$$

The log of this expression,

$$\boxed{logL_{max} = N^k(log\tfrac{N^k}{M^k} - 1)\ ,} \tag{20}$$

is the maximum likelihood fitness function for event data following a Poisson distribution.

In the other approach, we marginalize the likelihood in Eq. (15) with the finite-range constant prior, giving

$$P = \frac{1}{\lambda_2 - \lambda_1} \int_{\lambda_1}^{\lambda_2} \lambda^{N^k} e^{-\lambda M^k} d\lambda \tag{21}$$

yielding

$$P = \frac{1}{\lambda_2 - \lambda_1} \; \frac{1}{(M^k)^{N^k+1}} \int_{z_1}^{z_2} z^{N^k} e^{-z} dz \tag{22}$$

where $z_{1,2} = M^k \lambda_{1,2}$. In terms of the incomplete gamma function

$$\gamma(a, x) \equiv \int_0^x z^{a-1} e^{-z} dz \tag{23}$$

this is

$$\boxed{P = \frac{1}{\lambda_2-\lambda_1} \frac{1}{(M^k)^{N^k+1}} [\, \gamma(N^k + 1, z_2) - \gamma(N^k + 1, z_1)\,] \;.} \tag{24}$$

The unnormalizable flat prior that extends to infinity gives

$$\boxed{P = \frac{1}{(M^k)^{N^k+1}} \Gamma(N^k + 1) \;,} \tag{25}$$

differing slightly from Eq. (29) of [Scargle 1998] only because of different priors for λ.

Another commonly used prior is the so-called conjugate Poisson distribution

$$P(\lambda) = C \; \lambda^{\alpha-1} e^{-\beta\lambda} \;. \tag{26}$$

As noted by [Gelman] this "prior density is, in some sense, equivalent to a total count of α-1 in β prior observations" a relation that might be useful in some circumstances. The normalization constant $C = \frac{\beta^\alpha}{\Gamma(\alpha)}$ will be ignored. With this prior the marginalized posterior probability is

$$P = \int_0^\infty \lambda^{N^{(k)}+\alpha-1} e^{-\lambda(M^{(k)}+\beta)} d\lambda \;, \tag{27}$$

or

$$\boxed{P = \frac{\Gamma(N^{(k)}+\alpha)}{(M^{(k)}+\beta)^{N^{(k)}+\alpha}}} \tag{28}$$

Note that this prior and posterior reduce to those in Eqs. (28) and (29) of [Scargle 1998] for $\alpha = 1, \beta = 1$.

Equations (20), (24), (25) and (28) are the forms to be used whenever the counts in each tick follow the Poisson distribution. This includes both time-tagged data where duplicate tags are permitted and, as we will see below in §4.1.4, binned data.

Recently [Prahl 1996] has derived a statistic for event clustering in Poisson process data that tests departures from the known interval distribution (see the discussion above), by evaluating the likelihood over a restricted interval range. Prahl's statistic is

$$M_N = \frac{1}{N} \sum_{\Delta T_i < C^*} (1 - \frac{\Delta T_i}{C^*}) \ , \tag{29}$$

where ΔT_i is the interval between events i and $i+1$, and

$$C^* \equiv \frac{1}{N} \sum \Delta T_i \tag{30}$$

is the empirical mean interval. In other settings, the fact that this statistic is a global measure of departure of the distribution (used here only locally, over one block) may be useful in the detection of periodic, and other global, signals in event data. Results using the Prahl statistic are given below in §5.

4.1.2 0-1 Event Data: Duplicate Time Tags Forbidden

In this mode duplicate time tags are not allowed, the number of events detected at a given tick is 0 or 1, and the corresponding tick likelihood is:

$$L_m(\lambda) \quad = e^{-\lambda} = 1 - p \qquad n_m = 0 \tag{31}$$

$$= 1 - e^{-\lambda} = p \qquad n_m = 1 \tag{32}$$

where λ is the model event rate. From the Poisson distribution $p = 1 - e^{-\lambda}$ is the probability of an event, $1 - p = e^{-\lambda}$ that of no event. We can therefore use p or λ interchangeably to specify the event rate. Since independent probabilities multiply, the block likelihood is the product of the tick likelihoods:

$$L^{(k)} = \prod_{m=1}^{M^k} L_m = p^{N^k}(1-p)^{M^k-N^k} \tag{33}$$

where M^k is the number of ticks in block k and N^k is the number of events in the block.

There are again two ways to proceed. The maximum of this likelihood occurs at $p = \frac{N^k}{M^k}$ and is

$$L_{max} = (\frac{N^k}{M^k})^{N^k}(1 - \frac{N^k}{M^k})^{M^k-N^k} \tag{34}$$

Using the logarithm of the maximum likelihood,

$$\boxed{log(L_{max}) = N^k log(\tfrac{N^k}{M^k}) + (M^k - N^k) log(1 - \tfrac{N^k}{M^k})} \tag{35}$$

yields the additivity needed for our cost function.

An alternative way to quantify the fitness of the class of constant models to marginalize the rate parameter. That is to say, we remove this parameter by integrating it out:

$$P(B^k) = \int L^{(k)} P(\lambda) d\lambda \ , \tag{36}$$

where $P(\lambda)$ is the prior probability distribution for the rate parameter. Here we adopt a generic prior that is consistent with not having any particular prior information about the event rate, except that it must be positive. In [Scargle 1998] we used p as the independent variable, and chose a prior flat (constant) as a function of p. Here, we use a prior flat as a function of the rate parameter:

$$\begin{aligned} P(\lambda) &= \tfrac{1}{\lambda_2 - \lambda_1} \quad \lambda_1 \le \lambda \le \lambda_2 & (37) \\ &= 0 \quad \text{otherwise} & (38) \end{aligned}$$

The posterior, marginalized over λ is then:

$$P = \frac{1}{\lambda_2 - \lambda_1} \int_{\lambda_1}^{\lambda_2} (1 - e^{-\lambda})^{N^k} (e^{-\lambda})^{M^k - N^k} d\lambda \ . \tag{39}$$

Changing variables to $p = 1 - e^{-\lambda}$, with $dp = e^{-\lambda} d\lambda$, this integral becomes

$$P = \frac{1}{\lambda_2 - \lambda_1} \int_{p_1}^{p_2} p^{N^k} (1 - p)^{M^k - N^k - 1} dp \ , \tag{40}$$

with $p_1 = 1 - e^{-\lambda_1}$ and $p_2 = 1 - e^{-\lambda_2}$, expressible in terms of the *incomplete beta function*

$$B(z; a, b) = \int_0^z u^{a-1} (1 - u)^{b-1} du \tag{41}$$

as follows:

$$\boxed{P = \tfrac{1}{\lambda_2 - \lambda_1} [B(p_2; N^k + 1, M^k - N^k) - B(p_1; N^k + 1, M^k - N^k)] \ .} \tag{42}$$

The incomplete beta function reduces to the ordinary *beta function* for $z = 1$, so for the infinite range case $\lambda_1 = 0, \lambda_2 = \infty; p_1 = 0, p_2 = 1$ we have

$$\boxed{P\infty = B(N^k + 1, M^k - N^k) \ ,} \tag{43}$$

differing from Eq. (21) of [Scargle 1998] by one in the second argument, due to the difference between a prior flat in p and one flat in λ. All of the equations (35), (42), and (43), in their logarithmic form, can be used as fitness functions in the global optimization algorithm, and will be demonstrated below.

4.1.3 Time-to-Spill Data

As discussed in §2.2.3 of [Scargle 1998], reduction of the necessary telemetry rate is sometimes accomplished by recording only the time of detection of every Sth photon, e.g. with S=64 for the BATSE time-to-spill mode. This data mode has the attractive feature that its time resolution is greater when the source is brighter (and possibly more active, so that more time resolution is useful). The likelihood in Eq. (32) of [Scargle 1998] simplifies, with slightly revised notation and using the fourth comment at the beginning of this section, to

$$L^{(k)}_{TTS} = \lambda^{SN_{spills}} e^{-\lambda M} \tag{44}$$

where N_{spills} is the number of spill events in the block, and M is as usual the length of the block. With $N = N_{spills}S$ this is identical to the Poisson likelihood in Eq.(15), and in particular likelihood is at $\lambda = \frac{N_{spills}S}{M}$ and the corresponding cost function is

$$logL^{(k)}_{max,TTS} = SN_{spills}(log\frac{N_{spills}S}{M} - 1) \tag{45}$$

just as in Eq. (20) with $N = SN_{spills}$, and with the same property that the unit in which block lengths are expressed is irrelevant.

4.1.4 Binned Event Data

One of the most common data modes consists of counts in bins. The bins are typically predefined intervals in the measured variable. The count N_n in bin n is simply the number of values in it. The data cells are simply the bins and their associated counts:

$$\textbf{Cell} \quad \mathbf{n} \equiv \{\text{bin } n, N_n\}, n = 1, 2, \ldots, N_{total}. \tag{46}$$

Absent correlation effects, such as dead time, the probability distribution for the number of events of a bin is Poisson, and this data mode is equivalent to that discussed above in §4.1.1, with the bins taking the role of the ticks of that section. Here we generalize these results (and those in [Scargle 1998]) in two ways, allowing unequal bin sizes and a variable efficiency factor. The latter, sometimes called *exposure*, refers to anything that affects the count (*e.g.* instrumental sensitivity, dwell time, or uncorrected atmospheric effects). Assume that, whatever the nature of the effect, it can be represented by an efficiency factor between 0 and 1, such that the effective Poisson event rate is E times the actual (observed or modeled) event rate. Because of the nature of our piece-wise constant Poisson model, these two effects–bin size and bin efficiency–are equivalent in simply altering the local event rate, and can be represented with a single parameter equal to the product of the bin's width and efficiency.

The likelihood for bin n is found from the Poisson distribution:

$$L_n = \frac{(\lambda E_n W_n)^{N_n} e^{-\lambda E_n W_n}}{N_n!} \tag{47}$$

where λ is the actual event rate, in counts per unit time, and N_n is the number of events in the bin. The bin width W_n is expressed in the same units as λ^{-1}. The efficiency factor E_n is averaged over the bin. The product $W_n E_n$ can be replaced with a single quantity, $w_n \equiv W_n E_n$, expressing relative bin efficiencies.

The likelihood for block k is the product of the likelihoods of all its bins:

$$L^{(k)} = \prod_{n=1}^{M^{(k)}} L_n = \lambda^{N^{(k)}} e^{-\lambda w^{(k)}}. \tag{48}$$

Here $M^{(k)}$ is the number of bins in block k,

$$w^{(k)} = \sum_{n=1}^{M^{(k)}} w_n \tag{49}$$

is the sum of the bin efficiencies in the block, and

$$N^{(k)} = \sum_{n=1}^{M^{(k)}} N_n \tag{50}$$

is the total event count in the block. We have discarded the factor $(E_n W_n)^{N_n}/N_n!$ in Eq. (47) because, when multiplied out over all blocks in any model it produces a model-independent factor–its product over all bins. Any such common factor is irrelevant for model comparison.

Note that the block likelihood is essentially the same as that of Eq. (15). The only difference is that what we called a tick is now called a bin, and we have allowed for a bin efficiency factor (which in principle could be applied to ticks). Hence the maximum likelihood and marginal posterior cost functions to be used here are the same as those of Equations (20) (24), (25) and (28), with $M^{(k)}$ interpreted as the block-sum of the w_n instead of just the number of ticks in the block.

4.2 Measurements with Normal Errors

Here is a very common signal processing scenario: in order to estimate a signal embedded in noise, one makes measurements at a sequence of times. For example, if the noise is additive one has this nearly ubiquitous model for the time series observations:

$$x_n \equiv x(t_n) = f(t_n) + z_n \quad n = 1, 2, \ldots N \ , \tag{51}$$

where f is the unknown signal, z is the noise, and the observations times t_n may be evenly spaced or otherwise. We here consider the case where the noise is assumed to be normally distributed and with a known variance:

$$P(z_n|\sigma_n) = \frac{1}{\sigma_n\sqrt{2\pi}} \ e^{-\frac{1}{2}(\frac{z_n}{\sigma_n})^2} \tag{52}$$

The data cell then is denoted

$$\mathbf{X}_n = \Big\{ x_n, t_n, \sigma_n \Big\} \quad n = 1, 2, \ldots, N \ , \tag{53}$$

where x_n is the value measured at time t_n, and σ_n is the standard deviation of the noise.

In a block where the true signal is λ, the likelihood of measurement n is then

$$L_n = \frac{1}{\sigma_n \sqrt{2\pi}} \ \ e^{-\frac{1}{2}(\frac{x_n - \lambda}{\sigma_n})^2} \tag{54}$$

and the entire likelihood for block k is

$$L^{(k)} = \prod_n \frac{1}{\sigma_n \sqrt{2\pi}} \ \ e^{-\frac{1}{2}(\frac{x_n - \lambda}{\sigma_n})^2} \tag{55}$$

where the product is over all n such that t_n falls within the block. The exponential is the only factor that matters, since the rest contributes to the total posterior probability the constant factor

$$\frac{(2\pi)^{-\frac{N}{2}}}{\Pi_{n=1}^{N} \sigma_n}, \tag{56}$$

where here the product is over all N data points. Hence the block likelihood can be written

$$L^{(k)} = e^{-\frac{1}{2}\sum_n (\frac{x_n - \lambda}{\sigma_n})^2} \tag{57}$$

The maximum of this likelihood is found as follows: Clearly we can just as well minimize the quantity

$$Q(\lambda) = \frac{1}{2} \sum_n (\frac{x_n - \lambda}{\sigma_n})^2 \ , \tag{58}$$

which can be done by setting its derivative to zero:

$$\frac{dQ(\lambda)}{d\lambda} = - \sum_n (\frac{x_n - \lambda}{\sigma_n^2}) \tag{59}$$

so that

$$\lambda_{max} = \frac{\Sigma_n (\frac{x_n}{\sigma_n^2})}{\Sigma_n (\frac{1}{\sigma_n^2})} \tag{60}$$

Letting $\rho_n = \frac{1}{\sigma_n^2}$ be the weight corresponding to the variance of measurement n, and putting the resulting expression

$$\lambda_{max} = \frac{\Sigma_n \rho_n x_n}{\Sigma_n \rho_n} \tag{61}$$

into the log of Eq. (57), we have

$$logP = -\frac{1}{2} \sum_n \rho_n (x_n - \frac{\Sigma_n \rho_n x_n}{\Sigma_n \rho_n})^2 \tag{62}$$

$$logP = -\frac{1}{2} \sum_n \rho_n [x_n^2 - 2x_n \frac{\Sigma_n \rho_n x_n}{\Sigma_n \rho_n} + (\frac{\Sigma_n \rho_n x_n}{\Sigma_n \rho_n})^2] \tag{63}$$

$$logP = -\frac{1}{2} [\sum_n \rho_n x_n^2 - 2\frac{(\Sigma_n \rho_n x_n)^2}{\Sigma_n \rho_n} + \frac{(\Sigma_n \rho_n x_n)^2}{\Sigma_n \rho_n}] \tag{64}$$

$$logP = -\frac{1}{2} [\sum_n \rho_n x_n^2 - \frac{(\Sigma_n \rho_n x_n)^2}{\Sigma_n \rho_n}] \tag{65}$$

$$\boxed{logP = -\tfrac{1}{2}[\bar{x^2} - \tfrac{\bar{x}^2}{\Sigma_n \rho_n}]} \tag{66}$$

This expression is related to the *weighted variance*, although there seems to be no universal choice for how to define same. But it makes sense that the block cost function is this variance: the best constant model for the block should have minimum variance.

As in the other cases, we can alternatively marginalize λ, by choosing the flat, unnormalizable prior

$$P(\lambda) = constant \tag{67}$$

yielding for the marginal posterior

$$P(B_k) = \int_{-\infty}^{\infty} e^{-\frac{1}{2}\sum_n (\frac{x_n - \lambda}{\sigma_n})^2} \; d\lambda \tag{68}$$

Setting

$$a_k = \frac{1}{2} \sum_n \frac{1}{\sigma_n^2} \tag{69}$$

$$b_k = - \sum_n \frac{x_n}{\sigma_n^2} \tag{70}$$

and

$$c_k = \frac{1}{2} \sum_n \frac{x_n^2}{\sigma_n^2} \tag{71}$$

we have

$$P(B_k) \;= \int_{-\infty}^{\infty} e^{-\frac{1}{2}(a_k \lambda^2 + b_k \lambda + c_k)} \; d\lambda \tag{72}$$

$$= \sqrt{\frac{\pi}{a_k}} e^{(\frac{b_k^2}{4a_k}) - c_k} \tag{73}$$

The total posterior is of course

$$P = \prod_k P(B_k) \tag{74}$$

or, in terms of the additive log-posterior, we have

$$\boxed{logP = \Sigma_k \, logP(B_k) = \Sigma_k[-\tfrac{1}{2}\log(a_k) + (\tfrac{b_k^2}{4a_k}) - c_k]} \tag{75}$$

where the sum is over all blocks, k. This expression is defines a practical cost function for normally distributed data, as will be demonstrated below.

4.3 Distributed Measurements

The data can also consist of measurements of a quantity, averaged over a range of values of t – not at discrete point, as in the previous section.

An good example is the spatial power spectra computed from measurements of the cosmic microwave background radiation [refs.], where the different experiments have widely different *window functions* (the term used to describe sensitivity as a function of the independent variable – *i.e.*, spatial harmonic number in the CMB case). In this case the data array could consist of the structure in Equation (53) augmented by the inclusion of a window function, indicating the variation of the instrumental sensitivity:

$$x = \{x_n, t_n, w_n(t - t_n)\} \;\; n = 1, 2, \dots, N \;, \tag{76}$$

where $w_n(t)$ describes, for the value reported as X_n, the relative weights assigned to times near t_n, and all other quantities are as in Eq. (53).

This is a nontrivial complication if the window functions overlap, but can nevertheless be handled with the same technique.

We assume the standard piece-wise constant model of the underlying signal, that is, a set of contiguous blocks:

$$B(x) = \sum_{j=1}^{N_b} B^{(j)}(x) \tag{77}$$

where each block is represented as a *boxcar* function:

$$B^{(k)}(x) = \{ \begin{matrix} B_j & \zeta_j \leq x \leq \zeta_{j+1} \\ 0 & \text{otherwise} \end{matrix} \tag{78}$$

the ζ_j are the changepoints, satisfying

$$min(x_n) \leq \zeta_1 \leq \zeta_2 \leq \dots \zeta_j \leq \zeta_{j+1} \leq \dots \leq \zeta_{N_b} \leq max(x_n) \tag{79}$$

and the B_j are the heights of the blocks.

The value of the observed quantity, y_n, at x_n, under this model is

$$\begin{aligned}\hat{y}_n &= \int w_n(x)B(x)dx \\ &= \int w_n(x)\,\Sigma_{j=1}^{N_b}\,B^{(j)}(x)dx \\ &= \Sigma_{j=1}^{N_b}\int w_n(x)B^{(j)}(x)dx \\ &= \Sigma_{j=1}^{N_b}\,B_j\int_{\zeta_j}^{\zeta_{j+1}}w_n(x)dx\end{aligned} \tag{80}$$

so we can write

$$\hat{y}_n = \sum_{j=1}^{N_b} B_j G_j(n) \tag{81}$$

where

$$G_j(n) \equiv \int_{\zeta_j}^{\zeta_{j+1}} w_n(x)dx \tag{82}$$

is the inner product of the n-th weight function with the support of the j-th block. The analysis in [Bretthorst 1988] showns how do deal with the non-orthogonality that is generally the case here.[6]

[Note: the following repeats some of the above, and therefore needs to be rewritten.]

The averaging process in this data model induces dependence among the blocks. The likelihood, written as a product of likelihoods of the assumed independent data samples, is

$$P(\text{Data}|\text{Model}) = \Pi_{n=1}^{N}\,P(y_n|\text{Model}) \tag{83}$$

$$= \Pi_{n=1}^{N}\,\frac{1}{\sqrt{2\pi\sigma_n^2}}e^{-\frac{1}{2}(\frac{y_n-\hat{y}_n}{\sigma_n})^2} \tag{84}$$

$$= \Pi_{n=1}^{N}\,\frac{1}{\sqrt{2\pi\sigma_n^2}}e^{-\frac{1}{2}(\frac{y_n-\sum_{j=1}^{N_b}B_jG_j(n)}{\sigma_n})^2} \tag{85}$$

$$= Qe^{-\frac{1}{2}(\frac{y_n-\sum_{j=1}^{N_b}B_jG_j(n)}{\sigma_n})^2}\ , \tag{86}$$

[6]If the weighting functions are delta functions, it is easy to see that $G_j(n)$ is non-zero if and only if x_n lies in block j, and since the blocks do not overlap the product $G_j(n)G_k(n)$ is zero for $j \neq k$, yielding orthogonality, $\sum_N G_j(n)G_k(n) = \delta_{j,k}$. And of course there can be some orthogonal blocks, for which there happens to be no"spill over", but these are exceptions.

where

$$Q \equiv \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi\sigma_n^2}} \ . \tag{87}$$

After more algebra and adopting a new notation, symbolized by

$$\frac{y_n}{\sigma_n^2} \rightarrow y_n \tag{88}$$

and

$$\frac{G_k(n)}{\sigma_n^2} \rightarrow G_k(n) \ , \tag{89}$$

we arrive at

$$logP(\{y_n\}|B) = Qe^{-\frac{H}{2}} \ , \tag{90}$$

where

$$H \equiv \sum_{n=1}^{N} y_n^2 - 2\sum_{j=1}^{N_b} B_j \sum_{n=1}^{N} y_n G_j(n) + \sum_{j=1}^{N_b}\sum_{k=1}^{N_b} B_j B_k \sum_{n=1}^{N} G_j(n) G_k(n) \ . \tag{91}$$

The last two equations are equivalent to Eqs. (3.2) and (3.3) of [Bretthorst 1988], so that the orthogonalization of the basis functions and the final expressions follow exactly as in that reference.

4.4 Gaps and Mixed Data Modes

In some cases there are subintervals over which no events are possible (*e.g.* gaps due to failures in the detector system). What matters is the "live time" during the block, and this is simply the sum of the cell lengths. Thus data gaps can be handled by ignoring them! The only subtlety lies in interpreting what the model implies if a block extends across a gap. For each block the procedure yields the optimum rate parameter for whatever data lies in the block, ignoring any gaps. At the end of the procedure, for display purposes the gaps can be restored and plotted, preferably with some indication that rates within gaps are more uncertain.

Only if the fitness function depends on the total length of the block, and not just the live time, do the lengths of the overlap between the block and these gaps need to be included. The only example of this we have encountered results from the adoption of a prior distribution of block width.

Furthermore, one can even mix data modes. *E.g.*, bins of arbitrary sizes can be combined with point data. As with gaps the only burden for doing this is placed on the fitness function, which in this case would have to include a provision for data of mixed modes falling within the block. An example of this would be the analysis of both binned and time-tagged event (TTE) data for gamma-ray bursts observed by BATSE.

Which of the several posteriors above should be used? Should a new fitness function be constructed, based on ones understanding of the data and potential signals? If the conjugate prior is used, what values of its two parameters should be used? The answers depend on what is known about the data and its errors, as well as what one wants to assume about the signal. To aid in making such choices, §5.4 has relevant examples.

4.5 Prior for Number of Blocks

In our earlier work [Scargle 1998] no explicit prior probability was assigned to N_{blocks}, the number of blocks (or equivalently the number of changepoints). This omission amounts to using a flat prior, but in many contexts it is unreasonable to assign the same prior probability to all values. In particular, in most settings $N_{blocks} << N$ is *a priori* much more likely than $N_{blocks} \approx N$.

For this reason it is desirable to impose a prior that assigns smaller probability to a large number of blocks. Using a *geometric prior* for this parameter [Coram 2002] amounts to

$$P(N_{blocks}) = P_0 \gamma^{-N_{blocks}} \ . \tag{92}$$

The prior chosen affects the number of blocks in the optimal representation, a quantity of considerable importance since it affects the visual appearance of the representation and quantities derived from it. Most importantly, statistical fluctuations can be represented as real if large numbers of blocks are favored too much. While it is not a smoothing parameter as such, the effect of increasing γ can be mistaken for smoothing.

The form in Eq. (92) is not the only prior possible, but it is very convenient to implement, since with the fitness equal to the log of the posterior, one only needs to subtract the constant $log\ \gamma$ from the fitness of each block. Below in examples we show how the value of γ can be determined, and demonstrate that, especially with good signal-to-noise, the block representation is not very sensitive to the precise value adopted.

Figure 2 is the result of a simulation study using BATSE TTE data. The block decomposition of the full set of photons was taken as the (relative) truth and compared with decompositions based on a random subsample of the events. The RMS difference was taken as the measure of error, as a function of $log\ \gamma$ (the abscissa in the figure). This procedure is analogous to standard *crossvalidation* methods.

The effect[7] of $log\ \gamma$ in this and other simulations seems to level off

[7]A large value of this parameter naturally has the effect of reducing the number of blocks, producing a block representation that has less structure – giving a smoother visual appearance. But the parameter is not explicitly a smoothing parameter.

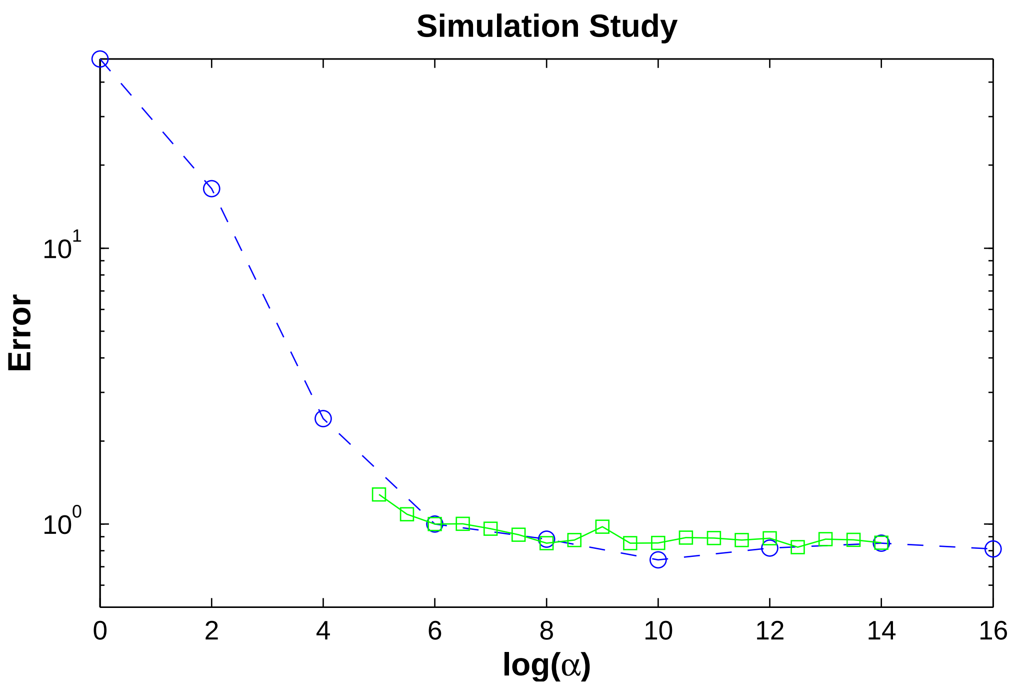

Figure 2: Simulation study for the parameter γ. The dashed curve with circles is for the first 110 busrts in the TTE sample, and the solid green curve with squares is for the complete sample of 1320 bursts, with DISCSC data.

at around 6. We have adopted the value 8 in the examples shown here. A simple argument, due to Mike Nowak, yields $\gamma \approx N$, where N is the number of data points.

These results are given not as a universal result for γ but because the general shape of the curve in Fig. 2 does seem characterestic of a wide variety of situations. We recommend that persons using the algorithm carry out simulations of this kind to study the behavior of the algorithm as a function of γ for their application.

5 Examples

This section presents results using the algorithms given in the Appendix on various sample data sets.

5.1 Determination of the Parameter γ

In applications, one must specify the prior for the number of blocks. The convenient geometric prior described in §4.5 amounts to the assumption that the prior probability of $k+1$ blocks is a constant factor, namely $\frac{1}{\gamma}$, times that for k blocks. Values of $\gamma > 1$ express the notion that a small number of blocks is *a priori* more likely than a large number.

In principle, the value of γ depends on one's prior knowledge of the number of blocks, but in applications it is rare that one can express this knowledge simply. In this section we perform block analysis of synthetic data where, knowing the correct answer, we can determine the best value.

5.2 Dynamic Range

One of the goals listed in §1 was that the algorithm have a large dynamic range. Here we give an example meant to demonstrate the dynamic range in both time and amplitude. The synthetic signal is a single block superimposed on a constant background, and the data are a set of points drawn from a distribution with the corresponding shape. The value $log(\gamma) = 8$ was used, and we adjusted the number of events in the spike to be as small as possible and still detect the spike. The errors in the block edges (-4 and +17 microseconds) are just perceptible in the figure. For as few as 4 events the spike was detectable

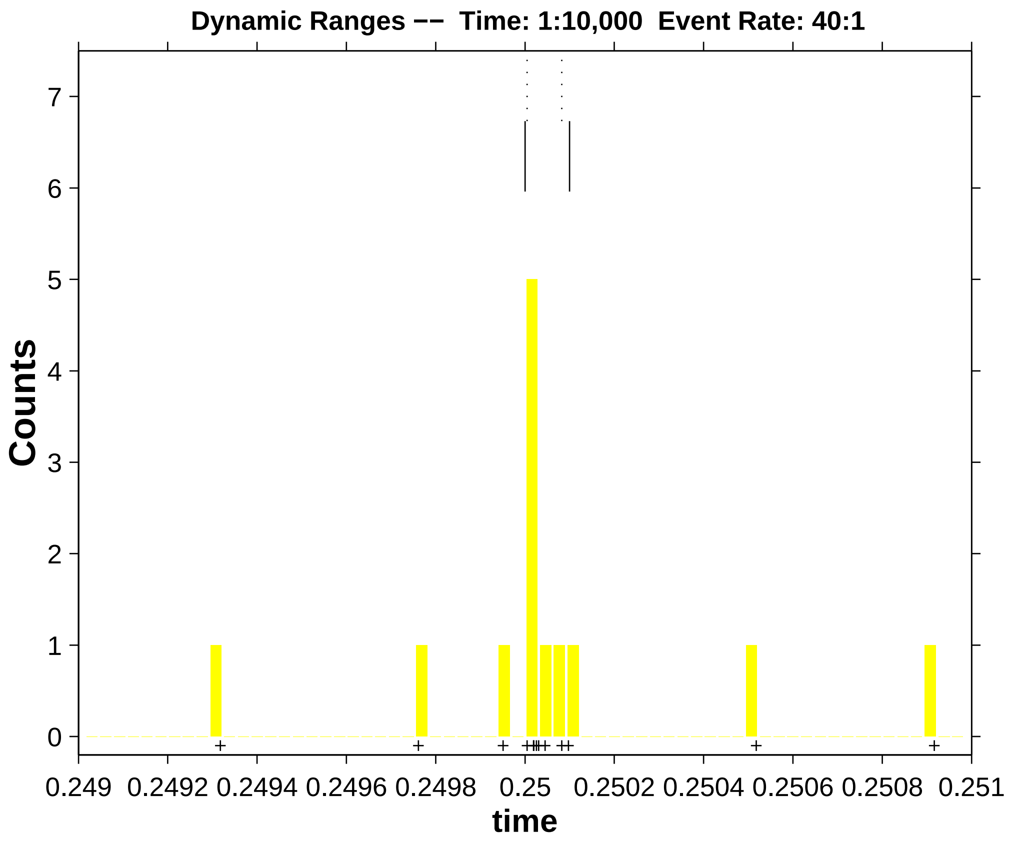

Figure 3: Maximum likelihood segmentation of a synthetic spike: 8 events in .0001 second on a background of 2000 events over the unit interval, only a small fraction of which is plotted. The solid lines at the top of the figure indicate the edges of the actual block; the dotted lines are the two changepoints of the optimal segmentation. The actual points are shown just below the histogram of the raw counts.

only by making $log(\gamma) = 4$, and with larger errors.

The next figure depicts a segmentation analysis meant to demonstrate the ability of the algorithm to handle a signal that has a large dynamic range in amplitude. The signal consists of three adjacent blocks on a small, constant background. The middle block has a much smaller amplitude, and the goal is to see if the near presence of large spikes on either side affects its edges. The rates in the spikes are roughly a million times the background rate and several thousand times the rate of the central satellite block. The dotted lines near the top signify the estimated block edges, or changepoints, whereas the solid lines denote the actual edge locations. The errors in the four edge locations are all less than 10^{-8} seconds. Our method is essentially impervious to large amplitude differences within a signal. In fact, increasing the number of counts in the main spikes in this example would only enhance the determination of the edges.

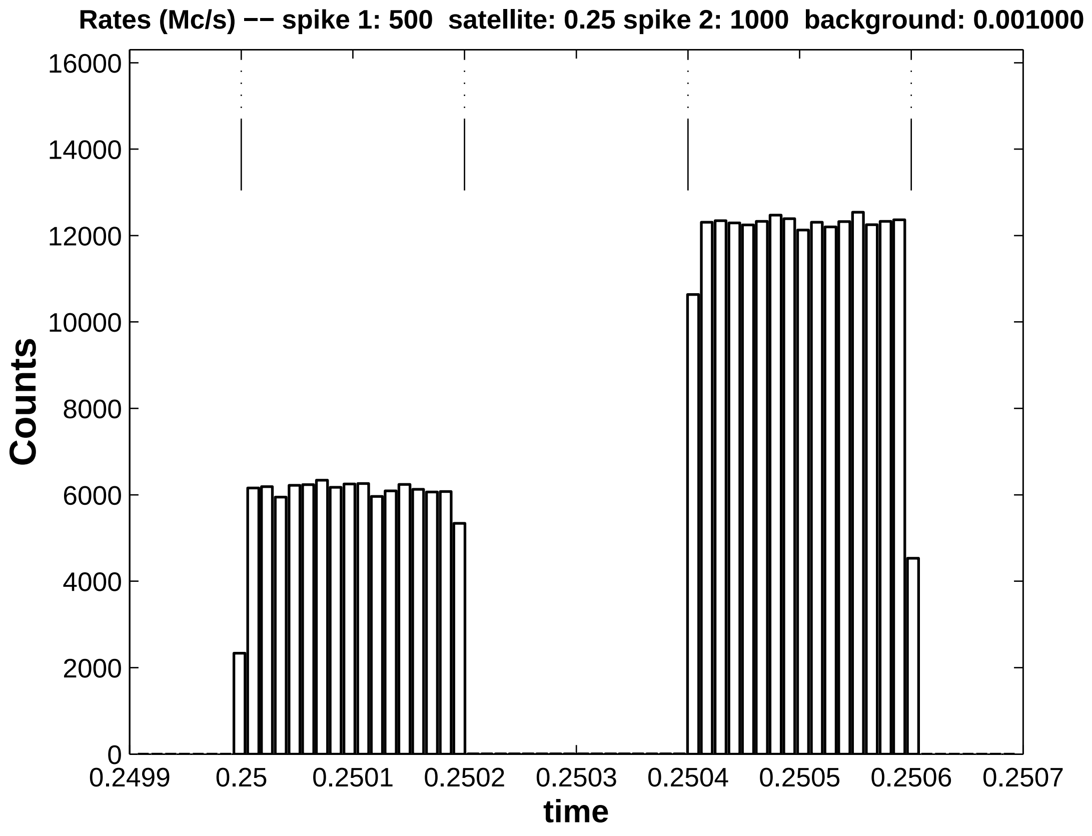

Figure 4: Maximum likelihood segmentation of a set of block with a large range of amplitudes. Each block has a width of 200 microseconds, with 100,000, 50, and 200,000 events, respectively, while the background consists of 1,000 events over the full 1 second interval analyzed. The central block and background are almost imperceptible on the scale of the figure. The analysis parameters are the same as in Fig. 3

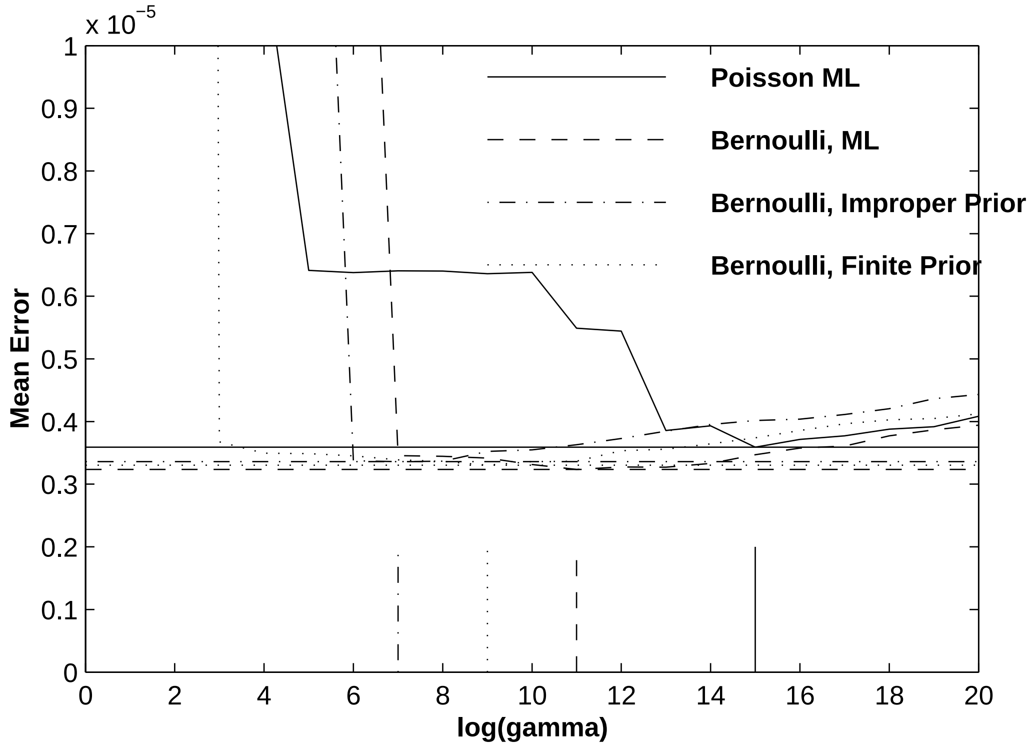

Figure 5: Simulation study, to find optimum value of the parameter logγ.

5.3 Point Data Time Series

This algorithm was originally developed with the BATSE TTE data in mind. Paper V used the greedy approach, which not only is not guaranteed to achieve the global optimum, but the iterative process that implements the greedy optimization requires a stopping criterion based on the adoption of a threshold. Even though it is possible to choose well-justified values for the threshold, this nevertheless represents an undesirable ambiguity. It is one of the nice features of the current algorithm that there is no such threshold.

Figure 6 shows the optimal block decompositions of data for a γ-ray

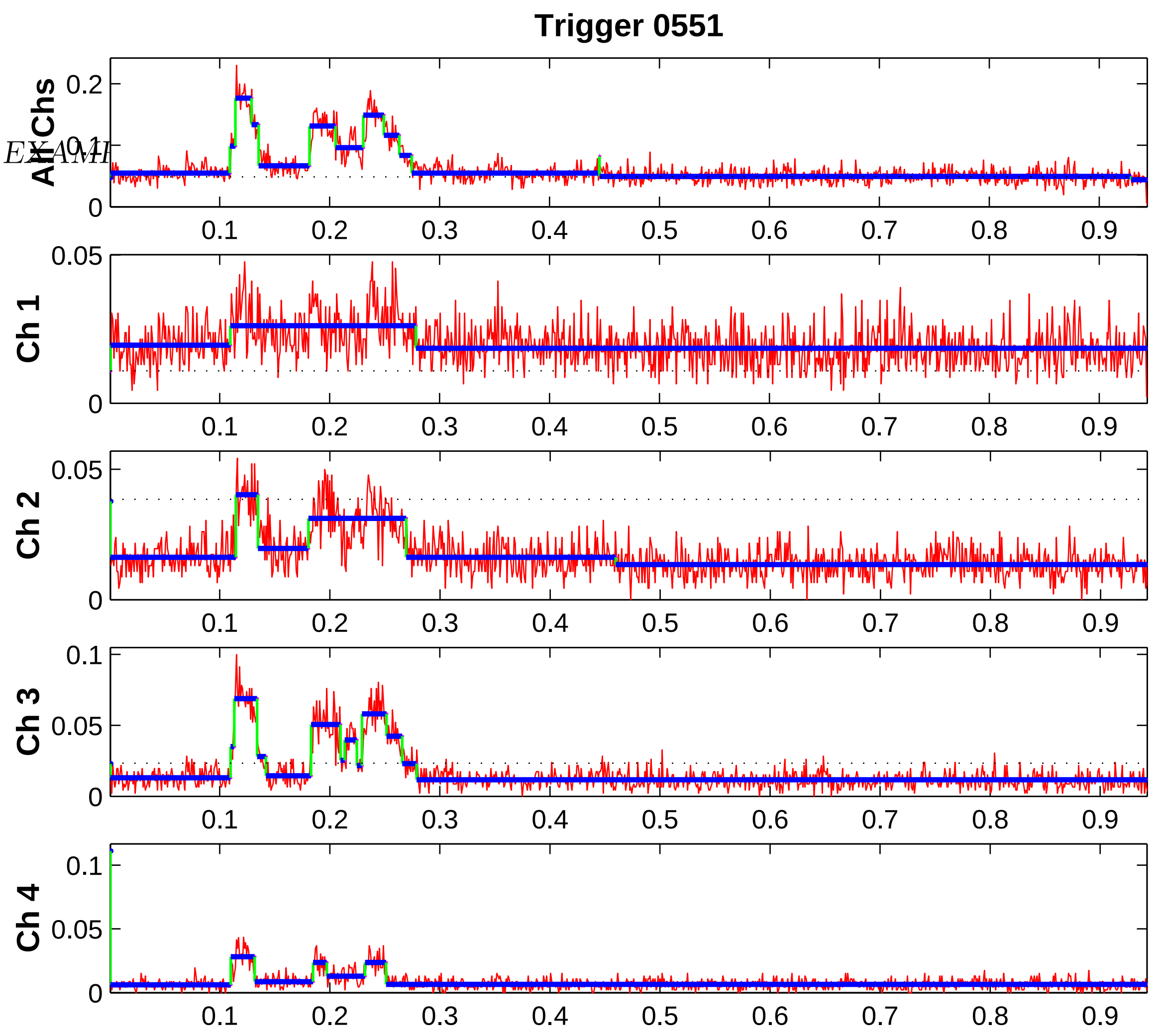

Figure 6: Optimal partitions of BATSE TTE data for Trigger 0551. All photons were used in the top panel; the others are based on the smaller number of photons detected in each of the four BATSE energy channels.

burst based on the point data comprising the TTE data for BATSE trigger 0551 (reference). The value $log_{10}(\gamma) = 8$ was used for the parameter in the prior. This analysis is based on the first $14,000$ photon time tags for this burst. The full data set consists of $28,904$ photons, but the last half is essentially background. Since the data are time tagged events, we used the form of the posterior given in Equation TBD. Need to compare duplicates allowed with not allowed.

Figure 7 shows the TTE data summed over all four energy channels, analyzed with four different values of the prior parameter $\log_{10}\gamma$. The first panel corresponds to a flat prior, giving too much prior probability to large numbers of changepoints. The obvious symptom is the appear-

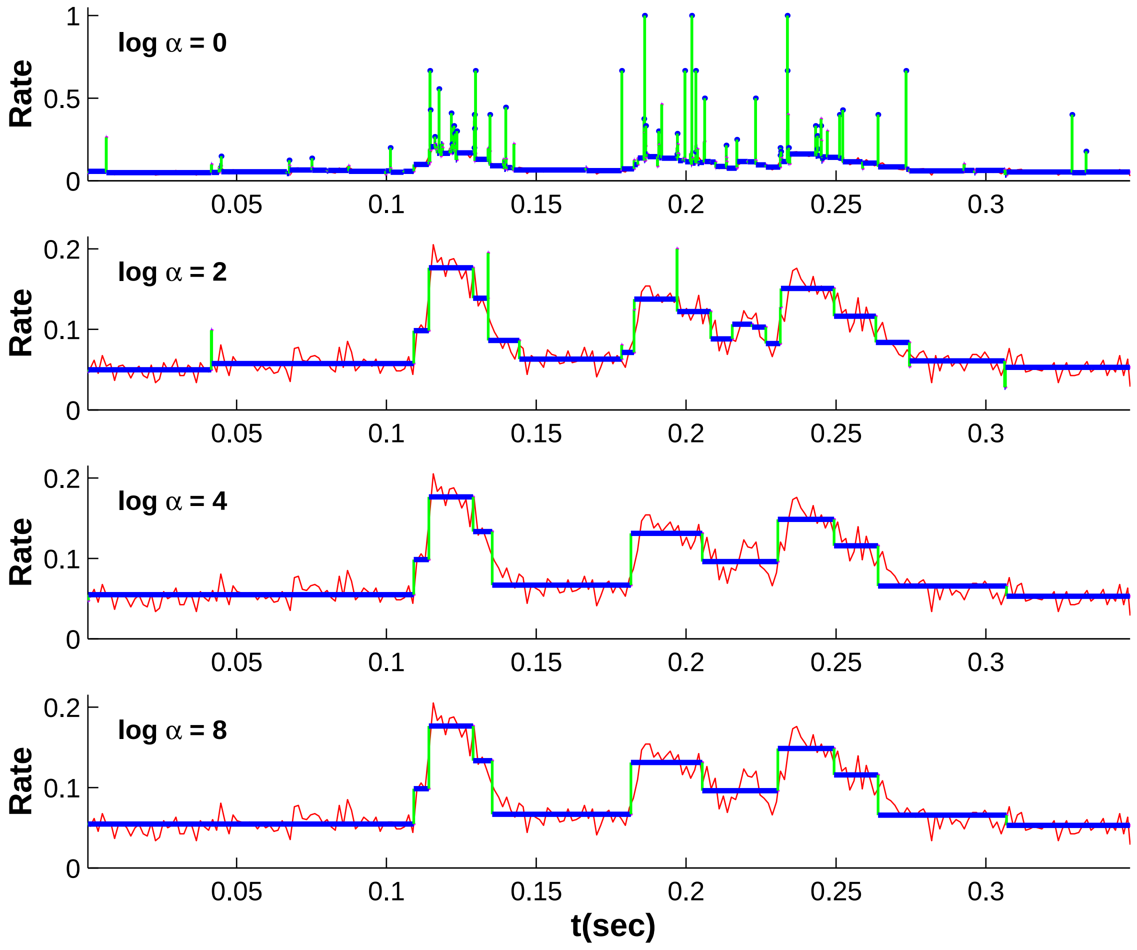

Figure 7: Optimal partitions of BATSE TTE data for Trigger 0551. Same as the first panel of Figure 3, except that four different values of $log_{10}\gamma$ were used: 0, 2, 4 and 8

ance of many short spikes, corresponding to narrow intervals in which statistical fluctuations are elevated by the inappropriate prior into apparent significance. While they represent putative features that are probably not real, and are cosmetically obnoxious, these spikes would not much affect the values of parameters derived from the curve.

The second panel, with a prior that gives lower weight to large numbers of changepoints has fewer spikes. By the time one reaches $\log \gamma = 4$, there is little change in the representation (*cf.* Figure 11). This result is not necessarily universal, but the figures shown here indicate that the value log $\gamma = 8$ is quite reasonable and that values somewhat lower or higher would not make any real difference in the final representation.

5.4 Binned Data

Figure 5 shows the block representation for a portion of the light curve of the first burst in the BATSE catalog, observed on April 21, 1991, Trigger 0105. These data are available [BATSE www site] [8] in binned format, with larger bins at the beginning, transitioning to smaller bins at the fiducial trigger time.

The three panels in the figure are for different values of the prior parameter log γ . The first case, log $\gamma = 0$, corresponds to a flat prior. With this rather strong encouragement for a large number of blocks, it is seen that the block representation is identical to the raw binned data. Even the coarse pre-trigger bins that seem to be combined into large blocks because their event rates are so similar, are represented as separate blocks.

The second panel, log $\gamma = 8$, corresponds to the best choice for the parameter, and can here be taken as the best block representation of these data. The last panel, log $\gamma = 16$, corresponds to too much of a penalty against a large number of blocks. One notes that the most intense peak, which is resolved into two peaks in the other panels, is here a single peak.

Finally, for comparison in Figure 6, we show analyses of the same data, unbinned and binned, for Trigger 0551. This figure was created

[8]BATSE continuously recorded data in time bins 1.024 seconds long, and the time series posted on the web has 116 seconds of such low-resolution data pre-pended to the 16 times higher (64 millisecond bins) resolution data starting at the fiducial trigger time. To make the bins equal, the numbers given on the web site apportion the counts in each large bin into 16 small bins. Since our analysis can handle unequal bins, we have undone this, and reconstructed the actual integer counts in the larger bins.

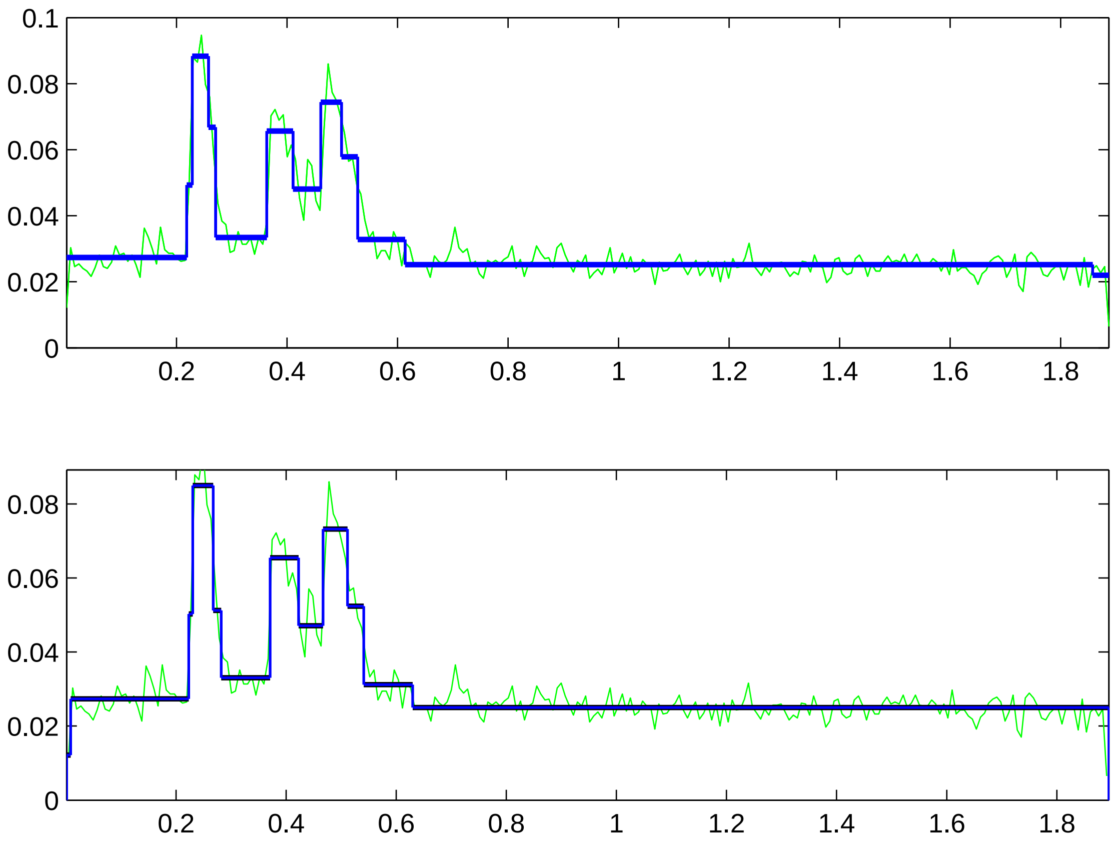

Figure 8: BATSE Trigger 0551. Top: TTE data. Bottom: same data, binned into 256 bins.

with the MatLab code included in the Appendix and available electronically. Note that the results are nearly identical, except for details of the first pulse.

5.5 Histograms

Histograms are perhaps the most used form of density estimation. Typically, bins are chosen somewhat arbitrarily, or perhaps large enough to have a "good sample," and to be equal in size, a choice that does not allow greater resolution where the data warrant it. The approach to this problem given here invokes the same procedure as described above for estimating optimal piecewise constant representations of point data. Unless the measurements are not independent of each other, the fitness functions derived above for point data are appropriate for histograms, which are by definition counts of events in intervals, and are governed by the Poisson distribution.

Often the data analyst is presented with a set of measurements, with little or no information about the measurement errors. In particular, the smallest measurable difference or quantum of the measurement (§4.1) is unknown. In such cases the maximum likelihood fitness function in Eq. (20) is appropriate, because its invariance property makes specification of scale of the measured variable or its quantum unnecessary.

The only item unspecified is then the prior on number of blocks (bins). Figure (9) depicts the results of a series of block determinations of synthetic data consisting of data distributed by a Poisson process which changes rate at known locations. The error of the representation can be readily computed by comparing the number and location of the actual and estimated changepoints, with the result that the optimum value of the parameter $log(\gamma)$ in the geometric prior can be determined. This kind of study is only valid for the specific problem simulated, but it is reasonable to assume that the results would not be drastically different for other situations. It is natural that the optimum parameter depend on the number of data points, and the figure shows the corresponding relationship.

The Matlab code provided in the appendix is essentially the same as that for point data, only with the natural choices for certain parameters hard-wired into the code. This is meant to be a turn-key algorithm for histograms, including a standard or recommended choice for the parameter $log_{10}\gamma$. Note that the algorithm, however, allows this parameter to be readily changed. It is recommended that any use of this algorithm be accompanied with experiments to determine how sensitive the results for that particular problem are to this parameter.

The Bayesian scheme in [Knuth 2004] for obtaining the optimal number of bins has a similar flavor to the current work, except that the bins are constrained to be equal in size. Another way to relax this constraint is to compute what are sometime called *equal-height* or *fixed-height histograms* – that is, demand that each bin contain the same number of data points. This of course leave the question of what this number should be. The current approach is meant to provide a principled way to assign bin sizes and locations as directed by the data, rather than by an *ad hoc* procedure.

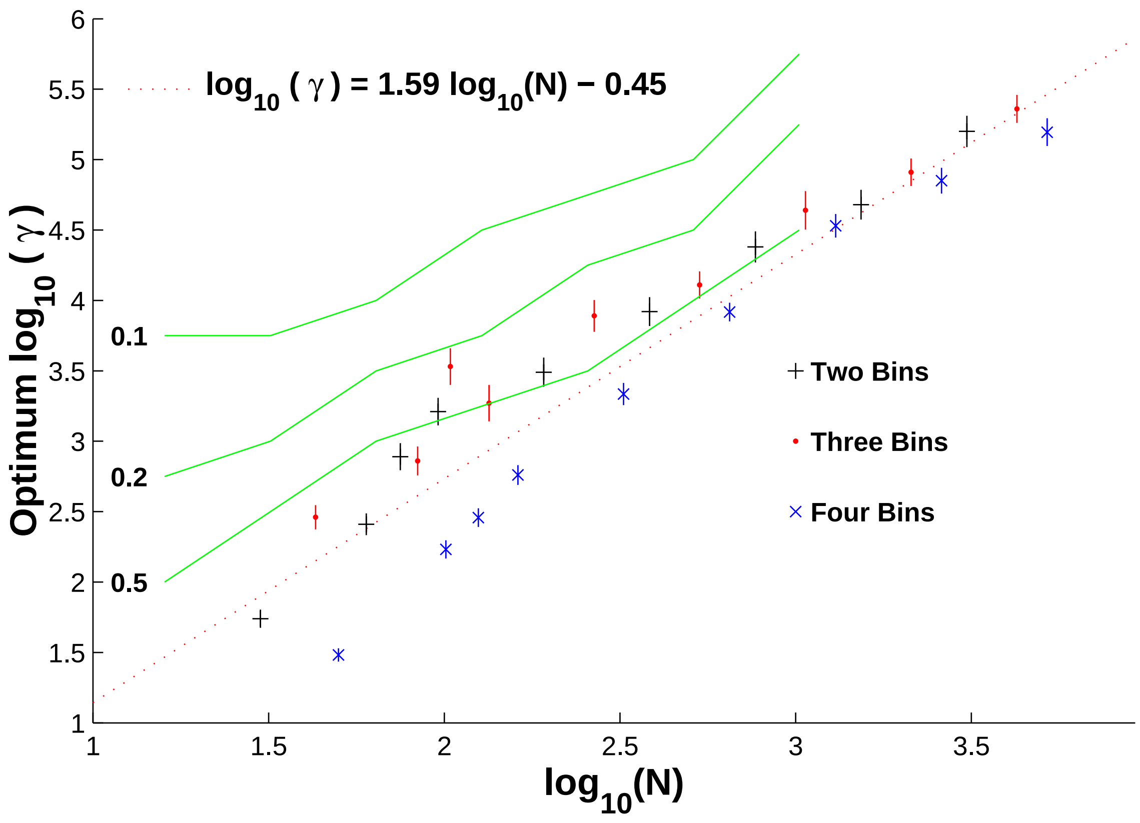

Figure 9: Simulation study: Optimum values of the prior parameter $log_{10}\gamma$ as a function of the number of data points N. 100 realizations were analyzed with a range of values of $log_{10}\gamma$; for each case, the optimum was chosen as that yielding the smallest error in the number and location of the bins. Different symbols are for signals with two, three, and four bins. The dotted line is a linear fit to the data for the latter two cases. A similar set of simulations carried out for the null case (a constant Poisson rate) are shown in the solid lines giving the value of $log_{10}\gamma$ that gives false alarms less than 10%, 20%, and 50% of the time.

5.6 Measurements with Normal Errors

First consider a simulation consisting of measurements at arbitrary times in an interval. These variates are taken to be zero-mean-normal, except over an unknown sub-interval where the mean is instead an unknown nonzero constant. Figure 10 shows synthetic data for three simulated realizations with different values for this constant. The solid line is the Bayesian blocks representation, using the posterior in Eq. (75). For the smallest amplitude (first panel), no changepoints are

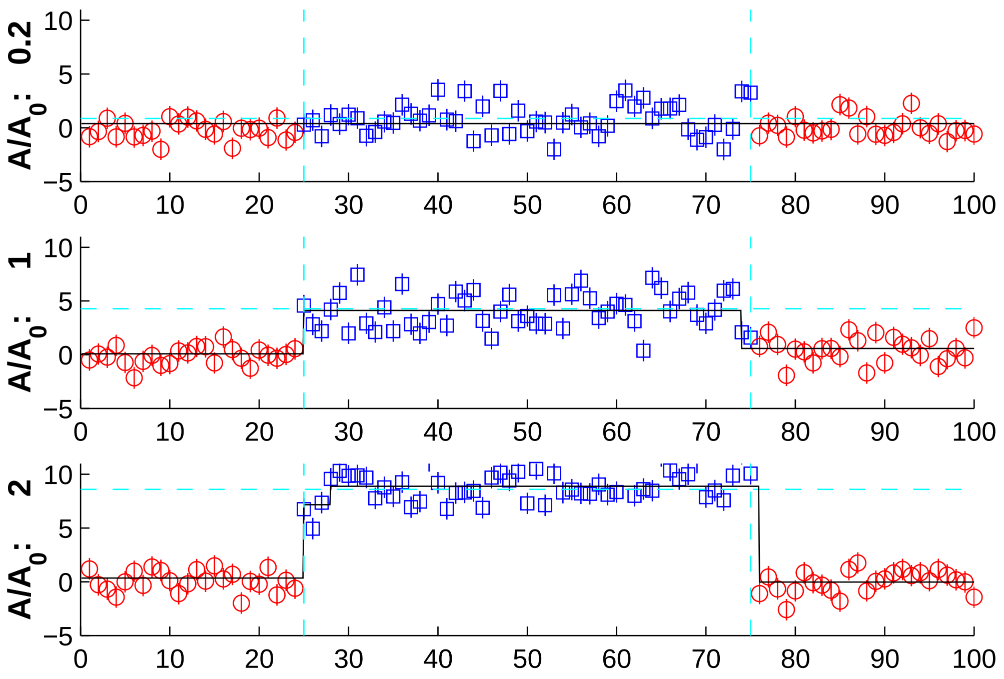

Figure 10: One hundred normally distributed measurements – zero-mean (circles) except for points 25-75 (squares), where the means are 0.2, 1.0 and 2.0 in units of the Arias-Castro *et al.* threshold $\sqrt{2\ logN}$. The dashed lines indicate the true changepoints and block amplitudes, and the solid lines are the Bayesian block representations.

found and so the signal is completely missed. In the next panel, the solution is correct except that the second changepoint is one point too early, while the solution in the third panel gets both changepoints correct.

In this experiment the points are evenly spaced, but only their order matters, so the results would be the same for arbitrary spacing of the data points.

A recent paper [Arias-Castro, Donoho and Huo 2003] on multiscale methods discusses essentially the same problem and develops several theorems for the aysmptotic behavior of optimal detectors of such signals. To quote these authors, “In short, we can efficiently and reliably detect intervals of amplitude roughly $\sqrt{2logN}$, but not smaller.” Fig-

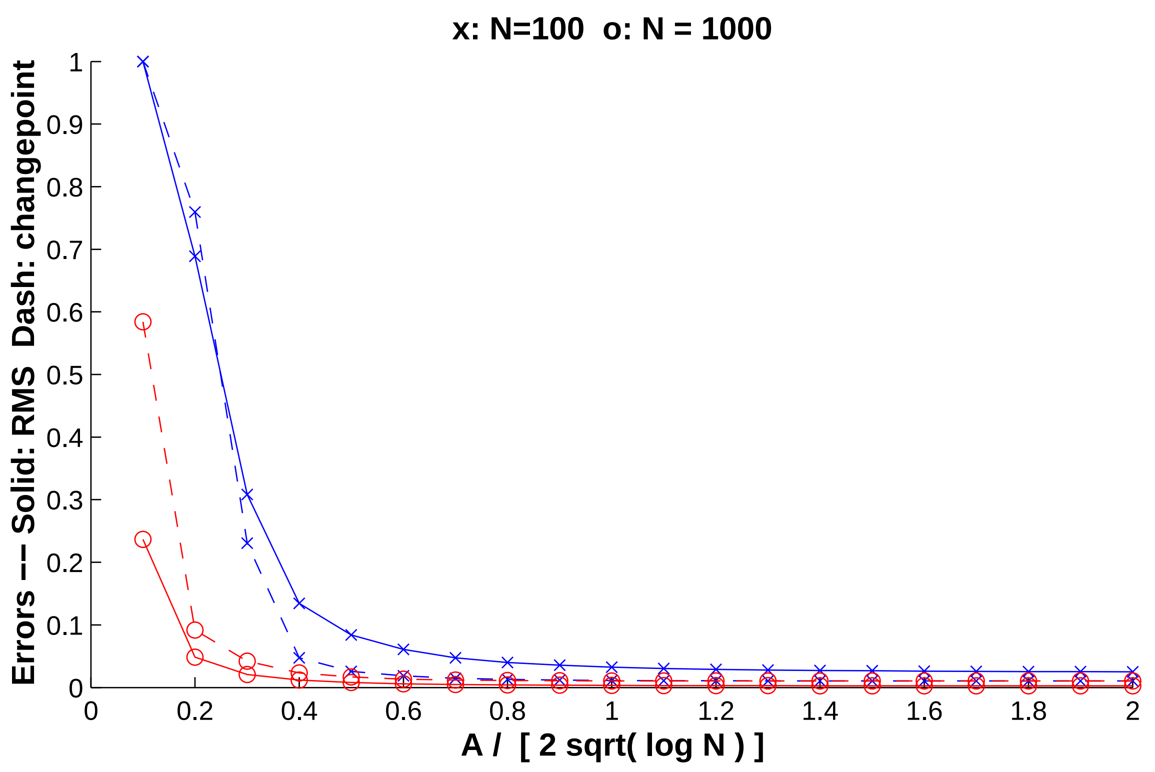

Figure 11: Relative error in finding a single block. **Abscissa:** True block amplitude in units of Arias-Castro *et al.*'s threshold amplitude. **Ordinate:** Error measures described in the text.

ure 4 reports some results of detection of the same normally distributed step-function process shown in Figure 10. The solid lines show the root-mean-square residuals from the true function, while the dashed line

This figure generally confirms this theoretical result, since the errors (both and a measure of the errors in the number and location of the changepoints) are

5.7 Real Time Analysis: Triggers

In many astronomical applications the task is to detect the moment when a signal rises above a presumed constant, or slowly varying, background level. The data stream usually consists of time-tagged or prebinned events. This detection problem is typically addressed by setting one or more event-rate thresholds. A signal exceeding threshold, *i.e.* producing a *trigger*, indicates the presence of a bursting or transient source from a region of the sky where previously none existed – or if it did, its brightness was below the threshold of detectability. Depending on the context, the primary goal may be to trigger as soon as possible, or with the greatest certainty (minimizing false detections or maximizing the detection rate for weak signals).

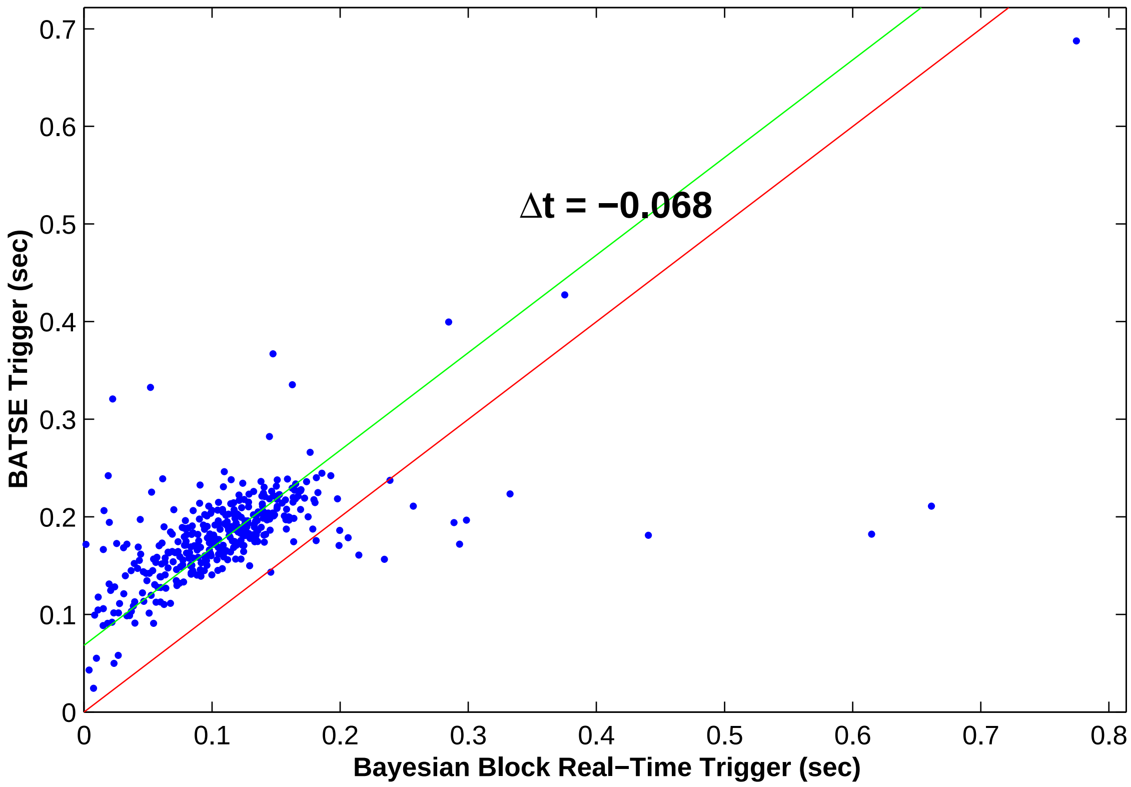

Figure 12: Direct comparison of on-board trigger times determined in real time by BATSE vs. those from the real-time mode of the algorithm described here. Points are shown for 367 short bursts where the BB representation consisted of at least two blocks, with the rate in the second block exceeding that in the first block.

The algorithm proposed here is simply the real-time mode of the optimal partition algorithm. It avoids the need to invoke event rate thresholds at all, and can be thought of as a scheme for detecting the first time the data present evidence for non-constancy, based on the know statistical distribution of the observational noise. Figure 12 compares the real-time triggers from the BATSE experiment (and recorded as part of the data structures available from HEASARC) with those from the real-time mode of our algorithm. Note that the BB trigger is systematically about 0.07 seconds earlier than the one determined on board the spacecraft.

A slightly more ambitious task might be to show the development over time of the structure of the signal – not just its first outburst. This differs from retrospective analysis of time series data only in that at a given time only the data up to that time are available. The estimated structure shortly before the end of the data interval at a given time may change, due the recently acquired data. Again, the real time mode of our algorithm can deal with this situation.

6 Appendix A: MatLab Code

This section contains MatLab[9] code for the analysis tools. The function `fit` evaluates the natural logarithm of the fitness function, and `reverse` reverses the order of an array. The quantity `eps` is the smallest number representable on the current machine. All other constructs and functions are standard MatLab.

6.1 Main Program

These code listings can be used to recreate Figure 6.

6.2 Construct Data Cells

6.3 Global Optimum

6.4 Load TTE Data

6.5 Logarithm of the Posterior Probability

6.6 Plot partitions

6.7 Plot TTE partitions

[9]© the Mathworks, Inc.

6.8 Reverse (from *WaveLab*)

This simple script for reversing row or column vectors is taken from WaveLab.

```
function r = reverse(x)
% reverse -- Reverse order of elements in 1-d signal
%  Usage
%    r = reverse(x)
%  Inputs
%    x     1-d signal
%  Outputs
%    r     1-d time-reversed signal
%
%  See Also
%    flipud, fliplr
%
   r = x(length(x):-1:1);

%
% Copyright (c) 1993. David L. Donoho
%
%
% Part of WaveLab Version .701
% Built Tuesday, January 30, 1996 8:25:59 PM
% This is Copyrighted Material
% For Copying permissions see COPYING.m
% Comments? e-mail wavelab@playfair.stanford.edu
%
```

6.9 Maximum Likelihood Histogram: ml_hist.m

The following script implements the maximum likelihood optimal histogram for data ordered and in the form of a MatLab row vector, as described in §5.5. It is much the same as the set of scripts for blocking point data, §4.1.1, except that it has been written as a single "plug-and-play" script. The only function that is not standard in recent versions of MatLab is *reverse*, an almost trivial function that reverses vectors (see above).

```
function [ change_points, rates ] = ml_hist( xx, ncp_prior )
% function [ change_points, rates ] = ml_hist( xx, ncp_prior )
%-----------------------------------------------------------------------------------------
% Compute optimal histogram of data: maximum likelihood partition of data interval
%
% Input:        xx -- input data, must satisfy these constraints:
%                      (1) must be a row vector ( 1 x N )
%                      (2) must be ordered
%
%        ncp_prior -- parameter in geometric prior on number of changepoints;
%                     optional [if not entered, default value
%                     1.59 log_10( N ) - 0.05 is used]
%
% Output:
%
%    change_points -- array of changepoint locations (index of xx)
%            rates -- array of event rates in the blocks/bins
%
%------------------------------------------------------------------------------------------

[ dum, num_xx ] = size( xx );

% The next 2 lines can be omitted if the conditions are known to be true:
if dum ~= 1; xx = xx';end  % If column vector, convert to row vector
xx = sort( xx );           % Order data

if nargin < 2
    ncp_prior = 1.59 * log10( num_xx ) - 0.45; % default value
end

%-------------------------------------------------
%   Put any identical values in a single cell
%-------------------------------------------------
cell_pops = ones( size( xx ) ); % Initialize: one datum per cell

for id = num_xx: -1: 2
    if xx( id ) == xx( id - 1 )
        cell_pops( id - 1 ) = cell_pops( id - 1 ) + cell_pops( id );
        xx( id ) = [];pops( id ) = []; % eradicate old cell
    end
end

%--------------------------------------------------
% Assign cell boundaries: data midpoints
%--------------------------------------------------

    dx =   diff( xx );
num_dx = length( dx );

cell_size = [dx(1)  0.5*( dx(1:num_dx-1) + dx(2:num_dx) ) dx(num_dx)];

%---------------------------------------------------------------
%                     Find optimal partition
%---------------------------------------------------------------

best = []; % "best(R)" is the value of the optimum at iteration R
last = []; % "last(R)" is the index at which this optimum occurs

%-----------------------------------------------------
%      Start with the first datum (R=1);
%      add the next one at each iteration
%-----------------------------------------------------

num_cells = length( cell_size );

for R = 1:num_cells

cum_cell_size = cumsum( reverse( cell_size(1:R) ) );
cum_cell_pops = cumsum( reverse( cell_pops(1:R) ) );

log_prob = cum_cell_pops .* ...  % Maximum likelihood cost
    ( log( cum_cell_pops ) - log( cum_cell_size ) - 1 ) ...
- ncp_prior;

[ best(R), last(R) ] = max( [0 best] + reverse( log_prob ) );

end

%-------------------------------------
%   Locate the changepoints
%-------------------------------------

index = last( num_cells );
change_points = [];
```

```
while index > 1
    change_points = [ index change_points ];
    index = last( index - 1 );
end

%------------------------------------
%   Count data in the blocks
%------------------------------------

num_cp = length( change_points );
num_blocks = num_cp + 1;
rates = zeros( num_blocks, 1 );

for id_cp = 1:num_blocks

   if id_cp == 1
      ii_1 = 1;
   else
      ii_1 = change_points( id_cp - 1 );
   end

   if id_cp >=  num_cp
      ii_2 = num_xx;
   else
      ii_2 = change_points( id_cp );
   end

   num_events = ii_2 - ii_1 + 1;
   delt_xx = xx( ii_2 ) - xx( ii_1 );

   if delt_xx == 0
      rates( id_cp ) = 0; % does not normally occur
   else
      rates( id_cp ) = num_events / delt_xx;
   end

end
```

7 Bibliography

References

[Bretthorst 1988] Bretthorst, G. Larry (1988), *Bayesian Spectrum Analysis and Parameter Estimation*, Lecture Notes in Statistics, Springer-Verlag. `http://bayes.wustl.edu/`

[Coram 2002] Coram, Marc, (2002), personal communication and Ph. D. thesis, Nonparametric Bayesian Classification,

`http://www-stat.stanford.edu/~mcoram/`

[Donoho 1994] Donoho, D.L., (1994), Smooth Wavelet Decompositions with Blocky Coefficient Kernels, in *Recent Advances in Wavelet Analysis*, L Schumaker and G. Webb, eds., Academic Press, pp. 259-308.

[Arias-Castro, Donoho and Huo 2003] Arias-Castro, E., , Donoho, D., and Huo, X. 2003, 'Near-Optimal Detection of Geometric Objects by Fast Multiscale Methods," preprint.

[Gelman] Book by Gelman.

[Hubert, Arabie, and Meulman 2001] Hubert, L., Arabie, P., and Meulman, J., 2001, *Combinatorial Data Analysis: Optimization by Dynamic Programming*, SIAM: Philadelphia

[Knuth 2004] Knuth, K. H., (2004), "Optimal Binning for Histograms," preprint.

[Kolaczyk 1996] Kolaczyk, Eric D., (1996), "Estimation of Intensities of Inhomogeneous Poisson Processes Using Haar Wavelets," Technical Report 436, Department of Statistics, The University of Chicago, Chicago, to be submitted to *Journal of the Royal Statistical Society, Series B.*.

[Kolaczyk 1999] Kolaczyk, E.D. (1999). Bayesian Multi-Scale Models for Poisson Processes. Journal of the American Statistical Association, 94, 920-933.

[Kolaczyk 2000] Kolaczyk, E.D. and Dixon, D.D. (2000). Nonparametric estimation of intensity maps using Haar wavelets and Poisson noise characteristics. The Astrophysical Journal, 534:1, 490-505.

[Kolaczyk 1998] Kolaczyk, E.D. (1998) Wavelet Shrinkage Estimation of Certain Poisson Intensity Signals Using Corrected Thresholds. Statistica Sinica, 9, 119-135.

[Kolaczyk 1998] Kolaczyk, E.D. (1997) Non-Parametric Estimation of Gamma-Ray Burst Intensities Using Haar Wavelets. The Astrophysical Journal, Vol. 483, 340-349.

[Kolaczyk and Nowak 2002] Kolaczyk, Eric D. and Nowak, Robert D., (2002), "Multiscale Statistical Models," Penn State Statistical Challenges 3

[Kolaczyk and Nowak 2002] Kolaczyk, Eric D. and Nowak, Robert D., (2002), "A Multiresolution Analysis for Likelihoods: Theory and Methods," preprint

[Nowak and Figueiredo 2002] Nowak, Robert D., and Figueiredo, Mario A. T., "Unsupervised Progressive Parsing of Poisson Fields Using Minimum Description Length Criteria," preprint

[Nowak and Figueiredo 2002] Nowak, Robert D., and Figueiredo, Mario A. T. (2002), "Unsupervised Segmentation of Poisson Data," preprint

[Okabe, Boots, Sugihara and Chiu 2000] Okabe, A., Boots, B., Sugihara, K., and Chiu, S. N. (2000), *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, John Wiley and Sons, Ltd., New York, Second Edition

[Ò Ruanaidh and Fitzgerald 1996] Ò Ruanaidh, J. J. & Fitzgerald, W. J., 1996, Numerical Bayesian Methods Applied to Signal Processing, Springer: New York.

[Papoulis 1965] Papoulis, A, 1965, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill: New York.

[Prahl 1996] Prahl, J., "A fast unbinned test on event clustering in Poisson processes," astro-ph/9909399.

[Scargle 1981] Scargle, J. (1981), Studies in astronomical time series analysis. I: Modeling random processes in the time domain. *Ap. J. Supp.*, **45**, 1-71.

[Scargle 1998] Scargle, J., 1998, "Studies in Astronomical Time Series Analysis. V. Bayesian Blocks, A New Method to Analyze Structure in Photon Counting Data", *Astrophysical Journal*, **504**, p. 405-418, Paper V. `http://xxx.lanl.gov/abs/astro-ph/9711233`

[Scargle 2001a] Scargle, J. D., (2001), Bayesian Blocks: Divide and Conquer, MCMC, and Cell Coalescence Approaches, in *Bayesian Inference and Maximum Entropy Methods in Science and Engineering,* 19th International Workshop, Boise, Idaho, 2-5 August, 1999. Eds. Josh Rychert, Gary Erickson and Ray Smith, AIP Conference Proceedings, Vol. 567, p. 245-256.

[Scargle 2001b] Scargle, J. D., (2001a), "Bayesian Estimation of Time Series Lags and Structure," Contribution to **Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering (MAXENT 2001)**, held at Johns Hopkins University, Baltimore, MD USA on August 4-9, 2001.

[Scargle 2001c] Scargle, J. D., (2001), "Bayesian Blocks in Two or More Dimensions: Image Segmentation and Cluster Analysis," Contribution to **Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering (MAXENT 2001)**, held at Johns Hopkins University, Baltimore, MD USA on August 4-9, 2001.

[Scargle and Babu 2002] Scargle, J. D., and Babu, G. J. (2002), "Point Processes in Astronomy: Exciting Events in the Universe," Chapter 20 of Handbook of Statistics: Stochastic Processes: Modeling and Simulation, 2002, Elsevier Science.

[BATSE www site] `ftp://cossc.gsfc.nasa.gov/compton/data/batse/ascii_data/64ms/trig00000/ca`